Engaging students with Computer Coding in the Visual and Applied Arts in Higher Education: The Possibilities, Challenges and Opportunities

Andrew Freeney

A thesis submitted in partial fulfilment
of the requirements of Liverpool John Moores University
for the degree of Master of Philosophy

July 2016

Abstract

The following “Expository” thesis paper is Postgraduate Research masters by research project entitled: “**Engaging students in the use of Computer Coding in the Visual and Applied Arts within Higher Education: The Possibilities, Challenges and Opportunities**”.

Although degree programs already exist in subject areas such as Computer Art and in Interactive Media there is a current trend that sees coding as a skill for all, and some even see it an emerging core subject along side subjects such as english and maths. As such there is opportunity for coding to be integrated in all areas of education, from Computer Science where it has its roots, to the Humanities and even Health and Medicine. This research looks at how computer coding can be integrated in a more universal form in the Visual and Applied Arts and highlights the important considerations and ways forward in todays new “coding for all” landscape, making it accessible and creating the opportunity for diversity through the integration of coding within visual arts subjects and indeed all subjects.

It is an original piece of research. Qualitative and Quantitative information was collected through the collection and analysis of data from questionnaires with students and through interviews with creative professionals who had a proven and extensive history in creative digital technologies and creative coding. It is a systematic investigation following a structured methodology. It builds upon previous working experience involving small focus groups of art and design students learning coding for the first time. It is designed for the purpose of expanding knowledge and understanding, in the implementation of new and emerging technologies within Visual Arts Higher education. It was a rigorous investigation in that much data was collected from both students and creative professionals. Questions and interviews were designed to draw out the most relevant and informative information possible. The significance of the findings lie in the potential to design modernised forms of teaching and learning using interdisciplinary technologies within the Visual Arts in Higher education. The research also outlines new avenues of enquiry which could form the basis of Doctoral Research.

Table of Contents

Chapter Title **Page**

1. Introduction

This research builds upon previous research; practice; teaching; and related qualifications gained in the field and relates directly to the three new organisational sections of John Moores University: Research; Teaching and Learning; and External Engagement.

Computer coding is reaching a point in its evolution where certain languages are simplifying and entering the public domain. It is more and more common for people to interact directly with the HTML code governing their own websites for example. Coding is more commonly seen in the prototyping phase of product design and it is also seen in the area of public experimentation or ‘tinkering’ along with Electronics.

Certain computer languages with their accompanying libraries have also been seen within Art and Design; as part of new media installations and electronic exhibitions. As more languages are being developed with beginners and Artists and Designers in mind there is now an opportunity to introduce Art and Design students to coding as a more fundamental concept. Art and Design Students do not have the same amount of time to learn programming languages as Computer Science students for example, but new, simpler languages may see more of them making use of coding. The term ‘Coding’ represents a recent rebranding of ‘Computer Programming’ previously seen as a set of complicated computer languages used to create the Information Technology Infrastructure that supports our Information Economy. Though tools like these have been used outside of this sphere, only recently have they become accessible to beginners with the advent of new language libraries and key terms that allow people to understand quickly and to engage in the coding as much or as little as they see fit.

It is hoped, by the coding community, the maker or ‘tinker’ community, innovation organisations such as NESTA, and the government, that more coders will be created in all areas of our economy leading to large research funds being created and allocated to the area. This is being reflected in Higher Education where there is also an interest in the development of Mobile devices such as in apps developed for student information. There is a current collaboration between a research lab based in the Cyprus University of Technology and University College Falmouth who are investigating the use of iPads within Art and Design. A more widespread knowledge of coding could arguably accelerate the uptake of these technologies.

Through working with students over the past few years I have met those who, in their final year, want to create projects incorporating computer coding but find that they lack the knowledge or time to do this successfully. Not surprisingly they see ‘Computer Programming’ as another language with a steep learning curve. This is especially true if the student has zero or limited previous experience. Without the proper support one set back can result in the abandonment of ideas that have the potential to lead to exciting and inspiring projects. This has lead me to want to better understand the landscape of coding technologies as well as common difficulties experienced by students, in order to attain a broad understanding of the subject. Through this I hope to create a framework for the support and development of coding languages within Art and Design, but this can only be achieved with the more fundamental understanding of the teaching and learning principles behind such a framework. As well as this the pitfalls, perceived or otherwise, need to be identified and avoided. This I hope to achieve with this Masters of Philosophy.

As stated in the abstract the originality of the research lies in both the methodology and the investigation into new and emerging technologies that have been created with artists and designers and coding beginners in mind. How these technologies can be adopted by teaching professionals as a way to harness computer coding in non computer science degrees, and the important considerations before implementation, are the main lines of enquiry. In this new “coding for all” landscape and with these new platforms for education and project orientated learning that come to light in the research, pathways and opportunities are identified to students in the visual arts, in higher education. The research investigates ideas regarding delivery of these technologies within higher education but also challenges conceptions of some of the structures of higher education and some of the themes that are present within it. The research involved a questionnaire with students of the John Moores University, in the School of Art and Design. The questionnaire was successful in the sense that the response rate was high and the questions were formulated to effectively gather information from the student body. The research also involved interviews with creative technology professionals from across many different creative fields but all with a relevance to creative coding. The structure of the interviews effectively drew out many salient points regarding the research topic and opened up many avenues of enquiry. The research also outlined many new avenues for further research which are covered in the conclusion.

2. Literature Review

2.1 Introduction

The Literature Review will be split into five sections. The current (post 2010) global climate with regards to computer coding will be outlined. This will start with the UK and American governments stance towards computer coding in education. This is being covered as they are having a large impact on public educational institutions through curriculum changes and through a more general agenda to increase the computer science and computer programming skill base within the knowledge economy. Secondly the "Global Learn to Code Movement" currently being backed by large private corporations, but also populated by many freelance, open source professionals, makers and amateurs, has been growing at a rapid pace in order to fill a skill gap both for large companies and also for everyday people as computer coding is arguably becoming a necessary basic skill as maths and English are. Thirdly will be an overview of the 21st century creative coders. As the two previous sections of the literature review represent two entities looking to increase coding knowledge across all sectors, this section will show the current frontrunners of computer coding within the Visual Arts. This aim is to highlight the potential of computer coding within the visual arts to begin to create a database of relevant artists for future academic activities. Finally the pedagogical requirements in terms of the teaching and learning of computer coding will be researched to highlight some of the problems students face when learning computer coding and more specifically how these will be interpreted in a visual arts context.

2.2 The UK and American Governments Policies Towards Computer Coding in Education

There is currently a reform of the UKs ICT curriculum underway in Britain. This is due to the fact that there has been criticism of the secondary school curriculum due to the lack of computer science and computer coding. Instead there is an overemphasis on ICT and secretarial skills using basic software such as the Microsoft Office Suite i.e. word processing, spreadsheets and databases. (Cellar-Jones 2011) These traditional Microsoft Office applications have little to do with the concerns of the creative artist but hold great potential for academic research and scholarship. (Greenberg 1991) Students have been arriving at university lacking the key skills that if they possessed could greatly accelerate their higher education and future job prospects. For example, the 2011 the Next Gen report that was published by the National Endowment for Science, Technology and the Arts (NESTA) highlighted that the visual effects and gaming industry was having to source talent from overseas because of skills shortages at home. (Livingstone, Hope et al. 2011) This was considered to be a failing of our education system from schools to universities and insisted a change for the industry to remain globally competitive. This was echoed by Google chairman Eric Schmidt who made his comments at the MacTaggart Lecture at the 2011 Edinburgh International Television Festival who said education in Britain was holding back the country's chances of success in the digital media economy and that he was flabbergasted to learn that computer science was not a standard part of the curriculum in UK schools. (Douglas 2011)

The gap in the UKs skills base, highlighted by NESTA received a reply from the UK government in November 2011. The government stated that a reform of the Teaching of ICT and Computer Science in schools was needed to better reflect the changing role of technology. (Great Britain. Department for Culture Media and Sport. 2011) This has resulted in the Rt Hon Michael Gove MP, Secretary of State for Education's public consultation on a proposal to replace the foundation subject of information and communication technology (ICT) with computing at all four key stages, earlier this year. (Great Britain. Department of Education and Science. 2013)

With a move towards having computer science including computer coding being taught as a key subject in secondary schools, we can expect more students to be entering university with basic knowledge in these areas across all disciplines, including Art and Design or, more specifically, the Visual and Applied Arts which will create more opportunities for students to enter the digital Avant-garde movement including areas such as generative art, algorithmic art, interactive or responsive environments, and wearables (electronics within fashion garments).

In the United States of America, similar changes are happening. In May 2013 Lauren Orsini wrote an Article entitled “Why Programming is the Core Skill of the 21st Century” and it highlighted the fact that currently in America lots of coding jobs were available for those with the appropriate skills and that having such skills would provide a tremendous boost to ones career (Orsini 2013). In 2013 President Barak Obama endorsed high school coding classes during a live Google+ Hangout. The president agreed that requiring students to learn a computer language made sense and he recalled a conversation he had with Mark Zuckerberg, founder of Facebook, about how he taught himself programming at a young age (Mack 2013). Even the mayor of New York City vowed to learn code via a twitter status update (Bloomberg 2012).

2.3 The Learn to Code movement.

The learn to code movement is growing at an exponential rate with names such as Bill Gates (Microsoft), Jack Dorsey (Twitter), Mark, Zuckerberg (Facebook) Drew Houston (Dropbox), Chris Bosh (NBA All Star), Will.i.am (Black Eyed Peas) and others all using and promoting computer code. (Partovi 2013) Hadi Partovi of Code.com had a short video film with these people listed above with others entitled “What most schools don’t teach” which opened by a quote from Steve Jobs who, when referring to the USA said “Everyone in this country should learn how to program a computer... because it teaches you how to think”. (Chilcott 2013)

There are a growing number of organisations promoting and providing free ways to learn to code online. These websites teach coding through various methods such as slides, presentations, wikis, video tutorials, games and quizzes. (Fox 2013) Two leaders in this who solely teach code online are Code Academy and Treehouse. The two are similar in that they both cover web technologies but Code Academy takes you through a few extras such as Python for example. A third example worth mentioning here is coder-dojo which is a global collaboration providing free and open learning to young people in locations across the globe.

In the last year, the MOOC or “massive open online courses” have spread through the higher-education landscape (Popp 2013). Online education giant Coursera, based in California, aggregates courses from 62 universities which are being taught online by professors and lets anyone take them for free. These are available in multiple languages including English, Spanish, French, Italian and Chinese. The Massachusetts Institute of Technology (MIT) has opened all of its course content to web audiences, so anyone, anywhere can learn from one of the top American research institutions. In addition to these, Udacity who have ported computer science, math and physics courses online have done so because they believe that education is lifelong experience rather than something that happens once in a lifetime. Finally Khan Academy is an online Academy that teaches everything from coding to calculus and computer science theory.

2.4 Key recent developers in the field of computer coding and new coding libraries

Programming, or code, plays a huge role in the world today. There is a growing community of artists whose artistic medium is code. Their work includes everything from computer generated art to interactive installations. Behind them is a growing community all with the goal of expanding our sense of what is possible with digital tools. (PBS Arts. Offbook. 2013)

Four key figures within this avant-garde movement, are Golan Levin (United States / Interactive Media Art), Casey Reas (United States / Creative Coding) Massimo Banzi (Italy / Open Source Hardware Hacking) and Zachary Lieberman (United States / Creative Coding). These four people represent key avenues in the pursuit of innovative digital art and provide a good starting point for the literature research. All have contributed greatly in their fields and are strong advocates of digital technologies and creative coding within art and design.

Golan Levin is an artist, composer, performer and engineer. Through performances, digital artefacts, and virtual environments, he creates interactive works which explore new modes of reactive expression (Levin 2005). In his paper Computer Vision for artists and Designers: pedagogic tools and technologies for novice programmers Levin attempts to demystify computer vision for novice programmers through a survey of new applications in the arts. Basic techniques of computer vision such as detecting motion and object tracking are discussed in addition to various software applications created for exploring the topic. (Levin 2006) After completing a Bachelor Degree in Art and Design at the Massachusetts Institute of Technology in 1994, Levin received a masters degree from the Aesthetics and Computation Group (ACG) at the MIT media lab in 2000. (MIT Media Laboratory Aesthetics + Computation Group. 2000) His thesis paper was entitled “Painterly Interfaces for Audio Visual Performance” and its goal was to develop an engaging new medium for audio visual self expression which is cited by many artists looking into digital performance art and audio / visual installations today. Notable artworks include Dialtones: A Telesymphony [2001], The Secret Lives of Numbers [2002] and The Dumpster [2006], Opto-Isolator [2007] and Double-Taker (Snout) [2008] (Levin 2015).

Casey Reas is an artist whose conceptual and minimal works explore ideas through the contemporary lens of software (Wikipedia contributors 2015). Reas was also a student in the MIT Media Lab's ACG graduating in 2001 together with Ben Fry, Reas developed the Processing programming language which is used by thousands of artists and designers worldwide (Reas. C and Fry, B, 2007). This is cited in numerous papers on programming within art and design including Programming for Artists with Processing by Zsófia Ruttkay (Ruttkay 2010). This paper highlights the two key reasons artists have problems learning programming. First: to acquire sufficient skill in computer programming and the essential algorithmic thinking is normally beyond the capacity of artists, and second: for many, it is not apparent what novel possibilities the computer offers for visual exploration and for entirely new genres of experience.

In his 2006 paper Reas, C. (2006). Media Literacy: twenty-first century arts education. AI & Society, 20(4), 444–445. Reas looks at a modern arts education system inclusive of digital and computational art. This is cited in two other papers by Kylie A. Peppler entitled Media arts: Arts education for a digital age, and Creative coding: Programming for personal expression. Peppler is an Assistant Professor in the Learning Sciences Program at Indiana University, Bloomington. An artist by training, Peppler engages in research that focuses on the intersection of arts, media, new technologies, and informal learning (Peppler 2015). Her 2013 paper (Peppler 2013) was a report for the Wallace Foundation in New York, which seeks to improve education and enrichment for poor children by testing innovative ideas (The Wallace Foundation 2015). Reas' works are numerous. Some recent works include “100% Grey Coverage (2013), Infinite Command Team (2013), Signal To Noise (2012), Yes No (2012)”. However there is an entire Process Compendium which is a catalogue of many works, too many to name here.

Massimo Banzi is the co-founder of the Arduino project which is a single-board micro-controller to make using electronics in multidisciplinary projects more accessible (Wikipedia contributors 2015). Through electronic sensors and environmental stimuli, like pressure, light, sound, movement events and interactions can be triggered for use in a visual or applied arts context. Banzi is an Interaction Designer, Educator and Open Source Hardware advocate. He started the first FabLab in Italy which led to the creation of “Officine Arduino”, a FabLab/Makerspace based in Torino (Banzi 2015). He is also author of the book Getting Started with Arduino (Banzi 2009) which is cited by many in the “Hacker” community including Alicia Gibb. Member of “NYCResistor”, co-chair of the Open Hardware Summit, and a member of the advisory board for Linux Journal, Gibb published the paper “New media art, design, and the Arduino micro-controller: A malleable tool” (Gibb 2010). In this, Gibb praises the Arduino micro-controller for its usability, supporting community and its open source initiative which empowers artists and designers to realise their creative ambitions. Banzi praised Reas and Fry’s Processing programming language highly saying “Processing changed dramatically the way we teach programming and it's one of the major factors of the success of Arduino.” (Reas and Fry 2010) Together with John Maeda, founder of MIT Media Lab's Aesthetics and Computation Group (and its replacement by his new Physical Language Workshop) he spend time working at the Interaction Design Institute Ivrea alone with key figures such as Bill Verplank and Bill Moggridge, who, together, coined the term Interaction Design. (Levin 2003) It was here that the Arduino project was conceived.

Zachary Lieberman is a digital artist and co creator of openFrameworks with Theo Watson. One of the key people behind YesYesNo, and an essential figure in the community of artists working at the forefront of media and technology, Lieberman is highly regarded by many in the field of media art. (The Creators Project 2015) openFrameworks is an open source C++ toolkit for creative coding. It was designed to assist the creative process by providing a simple and intuitive framework for experimentation. (openFrameworks 2015) Working with collaborator Golan Levin, he created a series of installations — Remark and Hidden Worlds — which presented different interpretations of what the voice might look like if we could see our own speech (YesYesNo 2015). This resulted in the paper In-Situ Speech Visualisation in Real-Time Interactive Installation and Performance (Levin and Lieberman (2004). Notable artworks include Night Lights (2010), iQ Font (2009), EyeWriter(2009), Reface (2007).

2.5 The Challenges of teaching computer coding

Computer Programming is becoming a more common skill and a valuable addition to any CV. However The increase in visual languages of all kinds used in multiple communication systems compounds the importance of art (Hicks, 1993). Research into the teaching of programming languages is extensive however few are based within the art and design environment. In Mateas' 2005 paper on procedural literacy, he states that all artists working in new media would benefit from the problem solving skills that are fundamental to programming i.e. logical, step by step thinking. Burg 2013 offers an alternative solution. The paper raises the idea that digital artists can create dynamic pieces without really knowing how to program. Programming environments exist which offer a pallet of tools for use in a drag and drop environment known as a stage. The "authoring" programs of Director and Flash are such environments where the code is generated behind the scenes and the artist is able to focus on the visual aspects of their project. The paper poses the question - does the artist really need to understand what's going on at the code level. Another aspect to take into consideration is that the Defence Advanced Research Projects Agency (DARPA), the american military's science lab launched a program in 2014 called MUSE. What this will be is a database of code snippets. The idea is that everything that anyone has ever tried to program will be loaded into this database which will act as a vast coding library. Instead of learning how to code people will teach computers how to think and the emphasis will not be on coding skills but on creative, problem-solving design thinking. Though this type of thinking has always been necessary when programming it will theoretically start to pull away from programming ability as the single essential skill when creating computer programs. In an article in Newsweek (Maney 2009) Kevin Maney states that computer coding is already a dying art. However it will take many years for the database to become fruitful and also to enter into the public sector and even then programming will take much longer to become obsolete but the idea of creative, problem solving skills being important is only going to increase as years progress.

Published in the paper "An interactive programming environment for enhancing learning performance." (Jambalsuren, M., Cheng, Z. 2002) there was the discovery that students often perceive programming languages as requiring significantly more work than other courses. This is reiterated in "Issues and difficulties in teaching novice computer programming" (Chan Mow, I.T. 2008) whose findings showed computer programming to be a cognitively challenging subject and that good instructional strategies are important in providing the student with optimal learner support. Different solutions to tackling the issue of complexity have been developed and are generally classified as technological pedagogical, or content based. Some of these have divided the elements of programming languages and rated them from a low and high level of comprehension in order to create instructional strategies to provide the student with optimal learner support such as in the paper "Computer programming and novice programmers" (Piteiral, M., Costa, C. 2012). Other interesting solutions include visually conveying programming concepts such as in, "Learning Programming Languages through Corrective Feedback and Concept Visualisation" (Watson, C. et al 2011), designing knowledge bases created for the purpose of analysing programs to see how they are constructed such as in "Design of a Knowledge Base to Teach Programming" (Weragama, D., Reye, J. 2012) or proposing authoring guidelines for the creation of learning materials for the computer programming novice such as in "Computer Programming Course Materials for Self-Learning Novices" (Okamoto, M., et al. 2010). In this paper inefficiency is highlighted as an issue and through my previous experience with teaching coding in Art and Design I have found that due to the complexity of the subject students can find themselves to be lost or confused. One possible solution to this inefficiency can be found in the paper "Designing a Constructivist-base learning environment: Using multimedia to engage students in a Malaysian classroom (Neo, M., et al 2010). In this research they introduce problem-solving and say that "Creative thinking, problem-solving, analysis and evaluation skills are very much needed in today's knowledge-based economy but still many graduates today are found lacking in them" and claim that this is resulting in a move towards more constructivist-based learning approaches. A lack of motivation is also an issue and in the paper "Learning motivation in e-learning facilitated computer programming courses" (Law, K., at al 2010) there has been the discovery that this can be improved through understanding 'individual attitude and expectation', 'clear direction', and 'reward and recognition'.

The move in computer science education since 2010 seems to be towards an interdisciplinary approach through using multimedia programming environments. Guzdial's paper (2008) is an example of how computer science professors are attempting to increase student engagement in the learning of computer coding through creating a more visually stimulating curriculum. This could be ideal for visual artists but here it is being looked at for computer science students.

This research aims to find ways to understand the implications of this changing landscape outlined in the literature review to increase the digital integration of coding specifically within the visual arts. It will look at ways of increasing motivation though possible outcomes and project based work. This could include marrying computer coding with other emerging technologies. Ways to make coding more appealing will be sought out. Computer coding does not tend to be a common skill in art and design so ways to maximise student engagement will need to be found. Transferable skills will be highlighted as well as ways to progress into more complex territory.

3. Methodology

3.1 Introduction

A primarily qualitative approach is being used within this research which will involve a case study with data collection. However some quantitative questions where added to the student questionnaires to ensure a broad spectrum of students were part of the research. Once a sufficiently diverse population was attained the decision to end the data collection could be justified.

More specifically, the collective case study will be used as I will be studying multiple cases in the single research study. This subjective approach has been decided upon as the opinions of all people partaking are being sought in order to achieve a holistic and descriptive view of the Creative Coding landscape. It is hoped that many different concepts and theories will be presented to explain how the research outcomes can be achieved.

The methods of data collection will be interviews with professionals within the creative industries, and questionnaires with visual arts students. The information gained will be combined with pedagogical theory regarding technology in the arts.

The case study is being employed as the roots of the case study are interdisciplinary and the research itself traverses computer science and art and design. The people being interviewed within the creative industries, for example, though all "technical creative practitioners", will inevitably learn more towards Computer Science or Art and Design but all will be bounded by the concept of the creative use of Coding.

The research aimed to look at coding in the "coding for all" landscape as mentioned in the literature review. Coding was being looked at as a universal skill and so looking across several disciplines aimed to capture data with this idea in mind. The scope of the research was limited to the Arts, and more specifically to the Visual and Applied Arts. This finite list of courses meant that although multiple disciplines where looked at, those disciplines were not worlds apart from one another, and in fact contained many close associations. A key part of the research was looking at how students who were not from a computer science discipline could engage with computer programming which is why the visual and applied arts was chosen. The research questionnaire was qualitative in nature. The three quantitive questions at the beginning of the questionnaire were added as a quality assurance measure to make sure a broad spectrum of student opinion was gathered. At this stage in the research the outcome was to identify a way forward that could cater for the entire student population of an Art School or Creative Institution. Further research and case studies into this area could then become more specific and look into specific discipline, age grouping, gender and other key identifiers. The year of study, specific discipline and level of experience was documented against the answers from the students and included in the appendix. This unique contribution to knowledge is available to be built upon with further research.

3.2 Interview Design

Interviews will be designed and arranged with programming and coding professionals in the Creative Industries. These types of interview are flexible (Bryman 2008). Though there will be a guide to the interview, the interviewees, and indeed the interviewer, will be allowed to deviate or even depart from the guide, in order to transcribe rich and detailed answers, capturing the interviewees' broader understanding of the Coding landscape within creative practice. There will be transcription of the interviews and subsequently analysis of the transcripts. The information collected combined with the information gathered from students, and the teaching and learning principles regarding technology in art and design will begin to triangulate a strategy for the digital integration of computer coding within the Visual and Applied Arts.

The following is descriptions of the professional interviewees who have been anonymised as per the ethical framework of the research. They have all signed the agreement (see appendix) which are not included in the submission but available for inspection.

- **Interviewee 1**

  The first interviewee is an expert on coding for the web, blogging, and content management systems. They co-curate an open source national coding festival and have extensive knowledge in hardware and software "hacking".

- **Interviewee 2**

  The second interviewee is a leading member of a creative coding arts collective. They are a prolific user of physical computing technologies. They have done numerous installations using coding and have extensive knowledge of other technological applications such as OSC, DMX, and MIDI.

- **Interviewee 3**

  The third interviewee is part of a new tech start up company. They are also an artist using creative coding. They run events for people interested in Processing. *Processing was made for Artists and Designers with an interest in generative and visual computer arts.*

- **Interviewee 4**

  Interviewee 4 runs a co-working space that focuses on Physical Computing, Creative Coding, Software Development, 3D printing, Laser Cutting and other forms of digital fabrication. *They contributed the IP addressing system for the Arduino board and run many workshops in the use of Arduino.*

- **Interviewee 5**

  Interviewee 5 is a long standing member of the new media art community. They have developed and delivered educational courses relating to the combination of technology and art in universities across the country. They are a prolific researcher into the areas of virtual space and telematics.

The interviewees will be encouraged to discuss what they feel Art and Design students should be aware of and what they remember as being problems when they themselves learnt different forms of coding. It is hoped that their opinions will form the basis of the strategies of teaching to non-computer science students. The rigid schedule of the structured interview was not chosen for this reason as here even the slightest deviation from the guide has the potential of unlocking these strategies and given complete freedom, greatly improves the chances of this happening.

The semi-structured qualitative interview has been chosen predominantly as there are already some questions that need answers; however the interviewees' points of view in what they see as relevant and important are of the utmost importance to this research, and so much leeway will be granted when the interviewee responds to questions to the point of an unstructured interviewing technique being employed, where the interview takes the form of a conversation (Burgess 1990). As there is no quantitative outcome there is no measurement to have its reliability or validity compromised (Bryman 2008).

Questions have been formulated to ensure depth is obtained. These will be injected into the conversation and with other follow-up, probing, specifying, structuring and interpreting questions as and when necessary. Silence will also be used as a means to control the pace and for reflection (Kvale 1996). Kvale (1996:133) gives us nine forms of questions, some of which are listed above which I will use as a framework. Questions were elaborated upon where necessary using the explanations in this section. The main introductory, direct and indirect questions in the semi-structured guide are as follows.

The first question to get things going was **“Can you tell me about yourself and your uses of coding in your creative practice?”** This created an easy starting point for the interviewee and allowed them to talk about themselves for a little while. As relevant points arose the interviewee was asked to elaborate and where natural pauses occurred contributions where made by the interviewee. Efforts where made to never talk over the interviewee and to welcome the information being volunteered. The aim here was to create a friendly collegiate conversation.

The next question was originally one of the final questions however it was moved into second place in order to add some context to the conversation early on. The question **“How do you think Computer Coding is perceived in Art and Design in the UK, and does there need to be a change?”** aims to map out the current climate in the UK with regards to Computer Coding in the Visual Arts, and to broaden the subject matter so that the conversation did not begin by going down a select or too narrow a pathway.

The third question aimed to look at how Art and Design Students could be assisted in engaging in computer coding, but this seemed to overlook a more fundamental question which is should Art and Design students learn Computer Coding at all. Although the answer may be obvious the rationale behind each of the interviewees opinions offers a wealth of information and knowledge to the interviewer and this type of open question could possibly reveal multiple avenues of investigation. So question three is **“Should Art and Design students learn Computer Coding, and why?”** Are there other alternatives for artists or other ways to fulfil their practice.

The answer to question three would also dictate the course of the rest of the conversation. Though it seems difficult to believe that this question would ever be answered negatively, such an answer would hopefully create important followup areas for investigation. However the goal of the research regards the best ways to integrate these technologies into Visual arts practice and so the next question becomes “**How can we get Art and Design Students to engage in computer coding?”** Due to the fact that experts in the Private Sector, Cultural Arts Practice, and Academia are involved in the research, many points of view will be gathered on the matter ranging from, experiences in teaching, experiences in learning, and experiences in practice.

The fourth question was originally "What kind of support do you think is needed; Teaching or otherwise?" however this question was considered too vague and also would not make sense to all of the interviewees and so question four was changed to **"What problems do/would you envisage art and design students experiencing when learning and using Computer Coding?"** This would reveal similar information and would allow interviewees to talk about their own teaching experiences or their own learning experiences and aimed to discover the answers to a question posed in the student questionnaire but from a different perspective.

The seventh question was **"How should Computer Coding be fed into art and design degree programs? What are the best methods of delivery?"** It was hoped that all the questions could be completely universal which was one of the reasons the previous question was changed however this was unavoidable as the context of the research directly related to Higher Education. It is acknowledged that this question may be problematic to those interviewees who do not work in education, However it is worth noting that all of them would remember their time at university and many of them still collaborate with universities and engage in some form of teaching and research on the matter. The question aims to look at different ways and different combinations of ways to integrate Computer Coding into a Visual Arts curriculum.

The next question is similar but worded in a slightly different way so as to get all available options to the surface. This question; **"What kind of structure would support students of different levels of proficiency and Computer Coding abilities?"** broadened the conversation to think about all students as a whole who come from different backgrounds and who will all have different abilities and levels of interest. Ideas generated by the last question regarding delivery may need developing or even changing in the light of this question.

Question 9 aimed to look at the relationship between the academic and the technician. The question is **"In your opinion how would the roles of the technician and the academic come into play in regards to this subject?"** The aim here was to look at how academics and technicians are perceived today. What is the role of the technician and what should they offer students. This question is particularly relevant in Art and Design practice where the technician can often work closely with the Artist or Designers to complete great works. Also Computer Coding is an inherently technical subject, being from the sphere of Computer Science so this question is particularly relevant in this research.

Following on from the last question the next logical question is **“How can we generate cross-departmental or interdisciplinary collaboration?”**. Is there an opportunity here to reconnect the Arts with Science. Can we develop a learning environment that has both types of students working together and sharing experiences. What are the best methods of achieving this and what are the possible outcomes.

The eleventh question was a topical one and one of the reasons the research was undertaken. It is regarding the change happening in secondary education that will naturally impact on higher education when the pupils arrive here. **“The UK secondary school ICT curriculum has faced heavy criticism for its lack of Computer Coding. What are the implications for H.E?”** The researcher is aware that soon we may have more students with basic computer coding ability. Is Higher Education ready to receive these students and take advantage of the skill base they already possess? Will they all have these skills or will we receive a broader range of skills. Will our curriculum be built to handle this?

A final catch all question was used in the form of **“Is there anything else you would like to add on the subject?”**. This question aims to uncover any ideas that the interview experience has created in the interviewees mind. Time will be given to allow the conversation to be adsorbed and for any new ideas or questions to rise to the surface. The researcher has the utmost respect for the interviewees choses and so much courtesy will be afforded to them during the interview process.

3.3 The relationship between the expert professionals and the student voices.

The research looked at two groups of people. The first was a group of creative professionals who both had extensive experience in their own personal and professional endeavours but also extensive experience in the visual and applied arts within higher education. This provided me with knowledgeable professionals within the context of my research. The second was a group of students from within the visual and applied arts. This provided me with the other side of the experience that was the focus of my research. Questions were formulated in a way to extract the most information possible from the first group creating a wealth of avenues of investigation within the subject of the research. The second group were asked related questions that were open ended and to a certain extend open to interpretation. They were also available online to be answered and anonymous. All these factors were to encourage engagement from the student population. The questions were however carefully designed to investigate and clarify the problems students faced, the perceived difficulties they had, the improvements they may want in their education, and the ideas they had for the future of the subject. It is hoped that concurrent themes can be identified and both sets of responses can be used to evidence one another and make the data gathered richer.

3.4 Student Questionnaire design

This first question aimed to find out how much students new about computer coding. The question was first posed as “what do you know about computer coding?”. It was decided that this was vague and abrupt. Also it had the capacity to alienate student who didn’t know very much which could have been a large percentage. The question was changed to **“As an Artist or Designer what are your thoughts on the use of computer coding?”** As an opening question this looked far more accessible and friendly to undergraduate students of all capabilities. Also it could capture what they thought of coding as well as some of what they knew about it. More information on their abilities could then be captured later in the questionnaire.

The second question aimed to build upon the first question by finding out some of the students’ abilities in the area. Again “what do you know about computer coding?” would have been to abrupt so the question was worded in this way. **“What experience have you had with computer coding?”** This allowed students who have not coded but who have seen it used to have a voice. It enabled students to talk about they’re indirect experiences as well as any coding they had actually done. It did not assume that the students had any coding knowledge which kept the question friendly and accessible.

The third question aimed to discover hurdles or pitfalls that students had experienced. The question “what difficulties do you have when learning computer coding?” was changed as this assumed that the students had already attempted coding and implied some failure on the students part. The question was changed to **“Has anything stopped you from learning or advancing in computer coding?”**. The question formulated in this way allowed students to talk openly about their experiences with coding including some of the problems they had faced, but placed no burden on the student to admit failure as something as simple as time constraints could have been a perfectly valid answer. The question is also designed to illicit answers from all levels of programmer.

The fourth question aimed to find out if students wanted computer coding in their curriculum. The question “Do you want to learn computer coding?” was a closed question and immediately abandoned. The question **“How much about computer coding, if any, would you like to learn?”** was used in the hope that it would illicit a cross section of responses from those who wanted a skill set for their CV to those who wanted to engage with it as a practice. It also allowed for multiple applications of coding to surface that students had been excited about and had the capacity to reveal new applications of coding thus undiscovered by the researcher.

The fifth question aimed to find out what students thought they could do with computer coding. The question “what can you do with computer coding?” was too broad, vague and demonstrated a lack of knowledge about what was being asked. The question **“What benefits do you see arising from gaining knowledge of computer coding?”** was used to find out what students thought they could do with this skill both on paper and in practice. It could be argued that they question is slightly biased, implying that there are only benefits to be gained from knowledge of computer coding, but the question was kept as the students could easily answer “none”, and, this research is based on previous experience in the field, and is looking to integrate this technology more fully into Visual Arts practice.

The last questioned looked to confirm problems that students faced with computer coding. Although the previous question i.e. “Has anything stopped you from learning or advancing in computer coding” touched on this subject the question was reformulated to get as much information as possible from the students as this is one of the primary investigative pathways. The question **“What problems do you think art and design students could experience when learning computer coding?”** was used as this made students not only think of themselves but of others. It was designed to make them consider their classmates and indeed students all around the country, and possibly the world, who are in a Visual Arts discipline and possibly struggling with computer coding. By collecting this outside perspectives it was hoped that the research would discover as much as possible about the difficulties Visual Arts students face with computer coding.

A final catch all question was used in the form of **“Is there anything else you would like to add on the subject?”**. This was aimed as those students with a burning idea or comment they felt did not fit into any of the other questions and also to those adept students who wanted to elaborate further on salient points or even just to demonstrate their knowledge, again, with the possibility of unearthing something which the research had overlooked.

4. Evaluation

4.1 Introduction

The following section will bring together a summery of the answers given to each question, it will also form some early conclusions based on the answers given. It is important to note that the questions were designed to deliberately overlap with one another so as to draw out as much information as possible. It was an open semi structured conversation and as such the answers to each question also overlap with one another.

4.2 Evaluation of Interviews

Interview Question 1.

How is computer coding perceived in UK and in art and design does there need to be a change?

The main point to take from the answer to question 1 is that the view of computer coding has changed to a certain extent and is continuing to change. Coding is broadly embraced and most people would want to see its use in Art and design education and in some creative circles outside of computer science it is totally accepted. A lot of people still don't see code as art and there could still be further change in how it is perceived. There is a school of thought that would say that the process of writing the code itself and debugging that code is an art form. Others would say what the code generates is the art. There are movements in the UK promoting the fact that code is art. Digital Technology and Art cannot be separated anymore. Digital Technology is too embedded in society. Everyone should learn computer coding, even art and design students. It is part of everyday life and is engrained in everything people do, so if you don't learn the language you are cutting yourself off. Too many people don't want to know how things work. They just want a box that you switch on. Code allows you to interface the real world with the digital world. It seems strange not to have new technologies in the classrooms. There has been a large improvement in Liverpool where this research is based especially now that organisations like FACT have had an impact on the scene. If you look back 10 – 15 years there were separate camps of "digital media art" but that has become much more part of art practice and people are much more switched on across all art practices and see digital art in some form as something to be integrated. This technology introduces you to open source environments which not only applies to the web but also to digital fabrication and interactive technologies. There are cross over opportunities here by getting computer science students to create the back end of a website for example and art and design students creating the front end. This would mimic the workflow in industry creating a graduate skill set that would be appealing to employers. The environment of computer science schools are however different to the studio or visual arts environment, this is what students are used to and changing this may not be conducive to their practice and their learning. There is a different learning methodology involved. A lot of students may be surprised by being faced with programming when leaving A-LEVELS. Students may be taken aback by that. This raises an interesting concept about the notion of self. People like others to perceive them in certain ways. If they model themselves as an artists sometimes they don't want to be seen associating computer science or with people in suits or capitalism for example. If people are generally artistic rather than being engineers they may have trouble in learning to think in the logical way that coding necessitates. Also digital Art is not always coded very well as artists are using a limited understanding of code. So any change that needs to happen needs to be tempered and carefully thought through before implemented. The power of online communities is an important one. There is a whole online world that can assist students, encourage students and help students to engage in new technologies and discuss their ideas with likeminded students or professionals in online forums. This creates much opportunity for development. The concept of time has changed now that people are constantly connected.

Interview Question 2.

Should art and design students learn computer coding and why?

If they're going to do digital arts then they definitely should however there comes a point where you need a better understanding of programming to accomplish advanced applications of computer code. The principles of coding, computing and networking are important to learn for any Visual Arts student. Not about any particular language, but knowing what it is and what the potentials are of it. Coding isn't just about sitting in front of a computer and crunching algorithms it is about systems. Coding and programming can open up different ways of thinking about things. If they can grasp what coding is and what computing is they would be much further on in their development. It should be an option.

Its the same as electronics. Collaboration is important with this learning. Play and experimentation is also important. The outcome is not the only consideration. The process is also important. To flip the argument around why would anyone make the argument to only use traditional media for art students. Older multimedia courses where just getting excited about having things move around on the screens or programming screen based interactions and now people are exploring the internet of things or maker based interactions and this still hasn't reached fruition in a lot of art schools. A key factor that should be of interest to Art and Design Students is that You can also make projects that are interactive or that do things. The student who comes to an art school now are different to students who came to art school 10 years ago because they are digital natives who have grown up with technology. Also the technological capabilities have totally changed. There needs to be two options. It needs to be taught in depth for those wishing to engage but also spread thinly throughout the curriculum to be used when needed. Coding needs to be taught in a way that prepares students for the real word with real word applications and does not simply teach it in isolation. The visual arts student can be given skills without the need to learn coding to the level of a computer scientist. Coding should definitely be an option but everyone should not have to do it.

Interview Question 3.

In what ways can we get art and design students to engage in Computer Coding?

There are tools that have been developed with this type of student in mind. Part of the reason the Arduino came out was to for the interaction design students to learn coding and electronics. It was made for artists and design but established at the Interaction Design Institute. (IVREA) It was made to teach the design students there who were very much graphical, product students and none “techie" with no formal coding tuition. Focusing development around outcomes and projects makes the goal outcome orientated. Then you can offer any kind of coding: coding written down on paper, coding that is sculptural, coding that is numbers and mathematics but you ultimately have to describe the outcome that can be discussed using the same methodology across these different outcomes. Graduate teaching programs are beneficial. Bringing back graduates to do sessions with students allows students to meet each other and share skills and get exited about what other people can do. Website creation would be a good way because its relevant to everyone particularly if you're an artist. If they had to do a bit of coding as part of this they may engage because they are getting something out of it. Its ok to copy and paste and use other peoples bits of code and hacking other peoples code to get used to it. However websites can also be limiting or even outdated. How often to you find websites that aren't updated or how often do you return to websites. Students could even build their own online communities and use code it that way. They could curate online gathering work together and displaying it under one banner.

Interview Question 4.

What problems do/would you envisage art and design students experiencing when learning and using Computer Coding?

Learning to code will require a significant investment of time. Students will need perseverance. You need to find supporting online communities. Everyones been where the new student is now. There is lots of documentation of other peoples attempts and mistakes. The more sophisticated you become to possible methods the more you can connect with online communities and discuss. The artist may struggle interpreting an array for example and how it works within computer coding. Its is a different way of thinking and another language. The student mat have difficulty in understanding how to get to the end product from a starting point of letters and syntax. Coding can be lonely, you need to be of a certain mindset to sit there and work on it. When you are trying to do a project with no background in engineering and coding you can be lost when you encounter a problem. That is why different approaches in both teaching and learning are necessary to engage the student. The teacher would be important. What happens when the student goes through the teachers lessons and comes back with a project that has already surpassed the level of the teacher. Opportunity to visit other departments may be necessary if you are dealing with electricity on an engineering level for example. Accessing online communities can be difficult. Knowing the etiquette of online message boards is a skill and questions need to be formulated in the right way to elicit the desired response. Also accessing other departments in university may be problematic. Does this require a level of agreement from all the departments that isn't really there in todays H.E. structure. Is there an adequate sharing of resources.

Interview Question 5.

How should Computer Coding be fed into art and design degree programs? What are the best methods of delivery?

Coding could be treated as just another material and the students could be exposed to the various application of it. Technology must be embraced but coding is not for everyone. Some people could be steered away from what they want to do and marginalised. It has to be available in a way that is comprehensive but also in a drop in studio environment. It will not be for everyone so anything curricular should be an elective. Students will react in different ways. Some will see it as easy. Some will see it as difficult but still want to engage and some will not want to engage but may or may not be swayed with adequate support.

Getting people more into using the web and exposing them to that. People don't get forced to build their there own website. Its useful for people to be online. This enables people to have online portfolios and it can make them more employable. Visual arts students and creative people can benefit from being visible to prospective employers online. Websites can be also outdated in the the sense that they are not always updated and not often returned to. So in a professional sense, as opposed to an employability sense i.e. for a professional artist and designer, other avenues need to be considered. The good thing about websites is the fact that can be online communities. Creative people can use the coding skills to develop their own online community. Curation online is also an interesting opportunity. People can gather work together and put it under one banner, or multiple students could co-author one single blog loping at a single topic. This would make the task of frequently updating the web easier as there would be multiple contributors. Students can learn by doing and be motivated in an outcome driven learning model. Student need to see the benefit from what they are learning. They can also use prefabricated Web Design environments to make things easier. So developing Wordpress themes, Drupal or tumblr can give them good starting skills and help them develop. They would not have to do the database coding which can be perceived as more difficult and instead design the front end or skin but still progress into more complex territory with javascript. This could be optional or be part of small credit modules. An intro to programming classes would be useful using simpler programming environments.

Interview Question 6.

What kind of structure would support students of different levels of proficiency and Computer Coding abilities?

This is important as a diverse student population is to be expected. Its important that the teachers are also practitioners as the use of technology is constantly evolving some people need to be onto of their field and actively engaged with it. There are always issues of staffing levels and resourcing. Good staff are key to supporting this. Not only within the subject but the communities surrounding the subject and the conferences, festivals, and open lectures around the country and the around the world. You have different levels of proficiently in both the students and the academics. Peer supported learning may be an option. Bringing back post graduate students could foster a creative community with creative coding. The community is key to allow people to learn from each other and develop their collaborative skills. Online learning and self directed study will also be important. Teaching students to interact with the wealth of support online could be a big factor in their development. Again, project orientated learning could be key to keeping students engaged. In a group scenario students could play to their strengths.

Interview Question 7.

In your opinion how would the roles of the technician and the academic come into play in regards to this subject?

Both have roles to play as it is quite hands on teaching. The academic could have more of a research background and be able to contextualise what is being learned. The role of the technicians in art is a complicated one. An artist has an idea but doesn't always know how to realise the idea so technicians can be brought in to realise the project. Traditionally the two sides where separate and not enough respect between the two roles. Often the technicians won't get the credit they deserve, some technicians do just give access to equipment but there are others which are more interested in their discipline. Then they are getting more towards being lecturers but that is an artificial distinction. There needs to be more crossover The roll of academic it to keep up to speed with the students and to know all the things that you can do with the subject at hand. But they should also know what is worth knowing and what isn't as you cant know it all. Technicians need to be more interested in helping people who want to learn and academics need to appreciate technical ability. This also points towards a new form of technical academic someone with high technical ability but who engages in research and scholarly activity.

Interview Question 8.

How can we generate cross-departmental or interdisciplinary collaboration?

Collaboration is complicated subject. It is very challenging to accomplish in a Higher Educational setting. IT can be an amazing thing, when the project can go in different directions to places you never thought it could due to input from multiple parties. Joint projects are the obvious place to start. Collaboration is not however the process of facilitating someone else's project. The term collaboration can be misused in this sense. Technicians for example are a facilitators as opposed to collaborators. If they collaborate then they become something different. Collaboration throws up a lot of questions about ownership and where the line is drawn. You need to understand what other people do in other disciplines. If you work in a team with people of a different skill set its important to know what they can do, so you know where the crossover point is. That relationship can be frayed and people can get disgruntled. Good relationships can be forged when both parties are working hard towards a common goal. Massive questions can come into play such as how are departments structured, how resources are allocated, what time management system is in place. Departments need to free time to allow for the collaboration or for other departments to come into their space. It looks very good on paper but its incredibly hard to implement. Is the H.E. model conducive to this kind of agreement? Does this require a level of agreement from all the departments that isn’t really there? Interdisciplinary interaction can take place from the fact that the people are there in the same room and are from many different backgrounds. This makes cross over and cross pollination possible through general conversation rather than any structured approach. There is no need to redesign the university from the top down but if space could be created for different approaches then this would be beneficial. Another option would be brokering. Putting together meaningful projects and bringing in student teams to complete various aspects. If you have public realm engagement and a real and meaningful output it can be a powerful motivator and also it forces you to reach a level of professionalism.

Interview Question 9.

The UK secondary school ICT curriculum has faced heavy criticism for its lack of Computer Coding, what in your view are the implications for H.E?

Its good that we are getting more programming in the curriculum. It has big leaps to go. When raspberry pi came out it started attracting more young people into hardware and software use, creation and hacking. We need to be ready to receive those students. Whether or not there will be compulsory programming lessons will be the question. If it is compulsory then that could be good because artists and designers would have experience however if it is not they may not elect to do it. The university may not necessarily receive more interested people but those who are interested in coding and the arts may be more skilled up. This may be beneficial to a university looking to build and support these activities. Art colleges and universities might not be ready for a new influx of computer programmers. There is a lot being talked about regarding this generation of digital natives. So universities may get more digitally literate but not everyone will learn to code. This idea that we will receive a tidal wave is unfounded. Education is shifting. It may not be shifting rapidly enough to create a massive change or as quickly as some interested parties would like. An interesting concept that is raised is that of knowledge. The generations growing up with the internet treat facts and knowledge differently. The internet provides us with a store of facts that we don’t necessarily need to know ourselves. They just make use of them.

Interview Question 10.

Is there anything else you would like to add on the subject?

Curriculum development and what we are facing in the future must be considered. It may be more community driven than sector – we can influence this through our research. That is the only way we can have sway over what happens. Open learning and "MOOCS" should come from the research community – Research is Key – Research community much retain influence on online learning .

The more the institutions become focused on teaching the less they will control what they are delivering and they will deliver more for the demand of the industry and student perception. Research must be influenced by industry but also shape industry. Identifying new creative industries, interface design, ludics. Don’t mass produce students – find where the next generation or new avenue of interest is going to be or lead.

Thankfully the world has changed. It didn’t used to be cool to do programming but now it is not seen like that. It is part of everything we do, smartphones, apps etc and people are wanting to know how to make apps. This may be another way in by making apps, frameworks exist that make cookie cutter type apps that people can then customise. This would be another good skill to take into industry.

4.3 Evaluation of Questionnaires

The questionnaires were returned by 40 students within the Liverpool school of Art and design some of which had graduated by the time of their response. The launch of the online questionnaire was timed in order to get the broadest cohort possible including both leaving graduates and new level four undergraduates in an attempt to capture the opinions in hindsight of the leaving students and the hopes and aspirations of new students. The answers to the first three quantitive questions are in the tables listed below.

Questionnaire Question 1

The students were split into the following degree programmes.

Course	Number of Students	%
Architecture	7	17.5
Fashion	4	10
Fine Art	3	7.5
Graphic Design and Illustration	17	42.5
Pop Music	1	2.5
Masters Architecture	5	12.5
Masters Fine Art	1	2.5
PGR	2	5

Questionnaire Question 2

The students were split into the following levels of education.

Level	Number of Students	%
4	1	2.5
5	5	12.5
6	11	27.5
Graduated No Further Study	15	37.5
Post Graduate Education	8	20

Questionnaire Question 3

The students were split into the following levels of proficiency.

Coding Experience	Number of Students	%
None	19	47.5
Beginner	18	45
Intermediate	3	7.5
Advanced	0	0

On first glance we can observe an imbalance in the response rate from graphic design and illustration. This needs to be taken into account when evaluating the results. The initial idea of gathering information from new level four students on their thoughts of computer coding in art and design has unfortunately failed as only one level four student has responded. This may be due to the fact that level four students may have yet to form an opinion on the subject or be exposed to the subject. This could be an opportunity for a further research paper. Gathering responses from leaving students and post graduate students has however been a success with 57.5% of the students responses falling into these categories. A good cross section of coding ability is represented which is pleasing. None fall into the advanced category which was expected and the majority of the students have none or are in the beginner level of experience. The questionnaire may have attracted students whom already have computer coding somewhere on their radar. A more random selection of students may provide different data. This may create the possibility of a biased data set but this does not discredit the questionnaire as it is possible that there could always be a divide between students who wish to engage with coding and students who do not. This will be addressed further on in the research.

Questionnaire Question 4.

As an Artist or Designer what are your thoughts on the use of computer coding?

From this point the questions, and indeed the main body of the research, are qualitative. Within the answers to the fourth question many positive comments where cited about the computer coding describing people who code as people to go to and even aspire to. It was acknowledged that coding is a new, underrated skill that you can do some truly amazing things with it. It was described as “Very beneficial” and a “very useful tool” that can be used to create your own creative tools. Some who had little understanding of what coding is knew they wanted to understand more, and appreciated that their practice could be enhanced by knowledge of it both through inspiration for new ideas and applications of practical skill. Some found it very enjoyable to engage with and where amazed at its capabilities.

The idea that coding is a current concept is acknowledged describing coding as something in the here and now. In this time where interactive computer technologies are pervasive, art should be a reflection of this modern, computer led time.

"I believe art ought to represent the time in which it is conceived. Computer coding must be considered a fundamental influence on the designer and, even if unused, ought to be acknowledged in the studio. Too often projects exhibit a superficial understanding of coding and in doing so undermine the value of ideas."

Even if it is not fully engaged with, it was noted that coding is not something that can be ignored. It needs to be tried to give students experience of designing and creating in different ways. It has its place in the art and design studio and it should be explored to see the different ways it can be incorporated into the students work. The idea of randomness and creating by accident is also an interesting concept that arises.

Employability and the broadening of skill sets is a key theme which arises when the students discuss computer coding. Some students see it as a way to get ahead of the completion giving them the edge in the jobs market. The Coding skill set compliments the artistic and design abilities of the students, pushing them ahead of the pack and making them more employable. Even if not excelled in some knowledge is seen as an enabler of collaboration by giving students the vocabulary to communicate with professionals in other sectors. Some only wish to possess low levels of experience in order to engage with web design tools such as Cargo and Muse which are adopting the prefabricated yet customisable template set out by companies like WordPress.

However, there are hurdles that the students perceive that need to be overcome. To some the language is alienating and coding feels worlds away from what some artist and designers might "usually do". Ways to be expressive with coding are not immediately apparent, only that it may creative something that might itself be expressive.

Some found it daunting, complicated and confusing in many aspects, and were only able to engage with it using simple applications or prefabricated examples. Lack of knowledge of what coding even is and how it can affect their practice is apparent in some, and lack of experience in others. It is acknowledged that basic principles are missing.

"I view it from afar with a mixture of curiosity and vague foreboding."

Although it appears to some students that coding seems to unleash endless possibilities, they are not sure exactly what these are and the whole thing seems out of reach. The perception that it is all too complicated is still present, and that only students and professionals in the Information Technology sector truly engaging with it. It's under appreciation is blamed on its steep learning curve and improvements in accessibility are seen as a slow process.

There is an increased pressure on designers to know some form of coding which may be exacerbating the problem of learning it. Some students see people in the art and design industry as being increasingly required to possess a multi-faceted skill sets, and are seeing computer coding ability appearing as a requirement on more and more job descriptions.

The students have identified some examples of ways they could engage with computer coding with web design leading these examples for the students own self-promotion and employability and at varying levels of expertise. However other interesting examples arise such as: the use of coding to create visuals for fashion catwalk shows; to program Arduinos which are micro-controllers made for artists and designers; the use of Isadora and Max/MSP which are visual forms of coding; the use of Processing which is a language designed for artists and designers to create graphical projects such as generative arts, which is also mentioned. Another interesting concept mentioned was to use code as a means to create your own digital tools and not be constrained by the computer applications that are made available to us by companies such as Autodesk and Adobe.

This is useful information as it shows paths that students are already engaging in. The research will look further into these technologies as well as other concepts that have been mentioned in the answer to this first question including: How we take this forward; there is still room for improvement in this area with a number of perceived difficulties: how do we get students to overcome these and engage; web design has been mentioned as an outcome driven option as have Arduino, Processing Isadora, and Max/MSP as good starter technologies. Logical thinking; It was mentioned that coding created a different way to design and create, and this is not necessarily something a visual artists may be used to. Creating the tools yourself; it was mentioned that students could be confined by the computer application that they use and that computer coding could be the real tool that allows them to express themselves. Employability was a reassuring theme. Students mentioned this being a valuable addition to their skill set. Learning by accident, or randomness illustrated a potential a better way to learn and to possible have the capacity to enhance the students learning experience. Crossing over disciplines; what technologies cross over disciplines and how can we generate interdisciplinary collaboration. Awareness and the student perception; Some students are still unaware of what coding is or the full potential of its application in the Art and Design Sphere. Others still see it as I.T. (Information Technology) and worlds away from what they do.

Questionnaire Question 5.

What experience have you had with computer coding?

Much interesting information was gathered from the answers to the fifth question. This was surprising considering the simplicity of the question and the complexity of the answers. Firstly some students cited experience in the technologies mentioned in the first question including coding websites Arduinos, and Processing as well as node based applications like Isadora, Quartz Compose and Module 8 (Max/MSP is also an example of a node based application). Most of the examples however cited coding for the web as the primary application so far. There was a range of uses of web coding technologies from students only “tweaking” CSS in content management systems such as Cargo, Joomla, Wordpress or other forms of blogs, to students coding full websites from HTML and CSS, students coding in JavaScript and jQuery which can make websites more transitional and interactive, and even students looking at languages that facilitate the back end of websites such as PHP. This is very encouraging information and shows that there are some students already engaging in many forms and levels of coding. Finally there was mention of C++, Java (of which processing is a derivative) as well as C# and ASP.net. Though this is truly impressive for an Art and Design student, it is hard to believe that this is the norm or even common among students of these disciplines. More investigation may be needed here. Three possible places where students have acquired learning have been identified.

The first being the university tutor, and the second being an online training facility, namely Code Academy. This shows another option for students to learn by accessing online resources and learning communities. Thirdly there was some mention of some learning happening before university. This is encouraging as it shows that some secondary schools are trying to push forward with coding and could be a sign of things to come with the current government initiatives to get more coding happening in secondary schools, something that universities obviously need to be prepared for.

A number of students simply listed “none” in their answers to question two. This could possibly be due to lack of awareness, lack of opportunity or not wanting to. This needs to be investigated further and may be elaborated on in later questions. Some answered none but acknowledged that they could see some potential in learning it.

“Absolutely none, but I would be willing to learn as I feel the digital world only opens more windows of opportunity”

Some students cited that coding could help them to display their work in different ways; could be used in employment such as digital media marketing; and could be used in creating interaction with a space, both by heightening the experiences within and by creating points of interest. One student cited needing a very good reason to engage with computer coding. This I find interesting as it points towards the idea of outcomes being key in generating student engagement.

More interesting themes are being discovered here including: Varying levels of engagement; students are discovering many ways to engage with coding ranging from the simple to the more complex. Allowing this flexibility has the potential to be inclusive of all students and enhance student satisfaction. Awareness or lack of interest; some students may not want to engage at all but some may just not be aware. We must increase awareness and allow student to make an informed decision regarding whether they want to engage or not. Outcomes may have the potential to drive students forward in this endeavour and websites seem to be the best contender with art and design practice as a close second. Clear outcomes need be identified to encourage this mode of learning?.Online training and leaning communities could provide a viable alternative support structure, these need to be assimilated into the university support structure. Coding can complement an art and design skill set and this opportunity needs to be used to its fullest extent.

Questionnaire Question 6.

Has anything stopped you from learning or advancing with computer coding?

The answers to question six elaborated on why students find it hard engaging. Having to learn a new, scary and alienating language was a barrier. The learning curve was steep and the inclusion of mathematics did not help. It was described as dry and frustrating and some did not know where to begin and found it difficult to learn leading to a feeling of being overwhelmed. A lack of immediate returns gave no positive reinforcement. A lack of a purpose, outcome, or practical application disinterested some students. A lack of time in university was cited again and again as the reason why students were unable to engage and outside of university they were unable to find support. Support was also limited inside the university as students were unable to access appropriate staff or other contacts. Some students simply did not want to engage as their practice involved more traditional methods and some have never had it offered or were unaware. A lack of experience before coming to university was seen as a missed opportunity. Having no prior theory or knowledge hindered student progress. It was pointed out that after a student returned on many separate occasions coding did begin to make sense. This is something I have realised myself. After a few attempts more and more was learned.

Some students interestingly mentioned no problems. They found motivation easy and learning resources to be accessible. Self-directed learning was evident here which is cited by one student as the key to success in this area.

" I would say that the principles of all software education is that one must have the responsibility to learn for oneself"

Other students required a more structured approach and found learning in a class to be more motivating. Inclusion in modules was cited as being required and an after-hours social group or club was seen as a good idea. Coding was seen by some as extracurricular and saw there university life as already too busy and unable to factor in something like computer coding.

Finally two interesting remarks are included here. One student mentioned a bad memory being the root of their problems and conversely one student mentioned their dyslexia as helping them get to grips with coding. This creates an interesting avenue of investigation regarding how the brain works.

All of this points towards a mixed approach as a possible method of delivering computer coding within art and design. Some modules could include coding and some after hours extracurricular sessions could also be offered. Taster sessions may be a good way of helping students with no experience before they have to deal with project driven or module based activity.

The themes being generated by question six include: inclusivity; how we help students with different abilities and needs. Diverse teaching; some students don't want to engage, some require structured module based teaching, some want after hours sessions or clubs and some are completely self-sufficient. Awareness or lack of interest; Some students are unaware of the possibilities and are unsupported in finding them. Some do not want to engage. Foundational knowledge; Some students are finding their lack of prior knowledge to be a hindrance. Programs need to be developed in order to help them to engage. Outcome orientated learning; Some students require a purpose or end goal in order to motivate themselves. Apparent difficulty; some students still find the learning curve into be too steep. This needs to be made easier. The threshold knowledge needs to be delivered to move student quickly passed the initial confusing stage ind into the visual developmental stage.

Questionnaire Question 7.

How much about computer coding, if any, would you like to learn?

The answers to question seven asked the students precisely what they wanted to get from using computer coding. Web design came up again and again in the answers to this question. This seems to be something the students are interested in, which makes total sense in the visual and applied arts where students have works that they wish to showcase. One student mentioned using the summer to achieve this showing that students are able to reach a level of self-sufficiency which is encouraging. The degree to which they want to understand web coding varied greatly. Some mentioned playing around with basic knowledge of HTML and CSS, the two standard front end technologies. Knowledge of these would allow you to build basic websites or utilise and personalise more complicated template frameworks such as WordPress. Some students wanted to go further mentioning building whole websites from scratch, some mentioned utilising JavaScript libraries like jQuery to add interactivity and some even want to go as far as engaging with back-end technologies such as SQL and PHP which creates databases, stores information and enables e-commerce activities such as online shops. This seems like a good skill set for an Artist and Designers.

Arduino and Processing were also mentioned. These two technologies were made for Artists and Designers in mind and remove a lot of the complexity associated with computer coding. The coding knowledge required is greatly reduced and the language and syntax itself is simplified. On top of this there is a vast online community of support exists to support the new learner making these two very interesting options.

Some students, though answering positively, where less prescriptive simply stating that they wanted to know as much as possible. One stated that they would need a good reason or outcome before they engaged. This shows a possible shift in the appreciation of coding within Art and Design. These students may realise that there are benefits from this knowledge or applications within their fields. Other students were looking to the future looking for any knowledge that could help them with future job prospects in their field. This aspect of employability seems to be a recurring theme as students appreciate the positive repercussions of a diverse portfolio of skills.

There was a group of students with more vague answers. They asked for a basic understanding of coding generally or, they had a basic understanding but wanted to know more, or stated that they wanted to know what benefits there where and what applications where within their field. The students who want to know more here, don’t state why or maybe they don’t know why. These answers all seem to point towards an awareness issue. One answer was very succinct.

“Would be interested in learning the capabilities and the possible outcomes of computer coding and to begin with the basics of how to go about putting something together to achieve an end result”

This answer in some ways shows one of the fundamental problems with computer coding. Students in the visual and applied arts may not immediately see how the letters on the screen translate into an end result which may take a variety of forms e.g. visual, physical etc.

Finally some individual students were very specific and diverse in their coding vision, mention applications such as creating games with interactivity, creating 3D objects for use within ArchiCAD, visualisation of music and projection mapping, learning JAVA, CC++ and C#.

The themes from question seven start to cement the recurring themes throughout the questionnaire write-up namely: employability; varying levels of engagement; awareness; diversity of teaching; and outcome orientated learning. More themes are further developing such as foundational knowledge and apparent difficulty in learning. Students need to be shown how the code can create works of visual art. The gaps in knowledge need to be filled to help students to make the mental transition from text to media. It seems to be accepted that a shift is happening in the perception of coding within art and design which is becoming more accepting. This shift will be fuelled if a change in the secondary school curriculum sends more students to university with basic coding skills. Self sufficiency or self directed learning seems to be a way to lessen the impact and also to create ambitious students. Online communities offer a support mechanism for the technologies that are not yet fully integrated into the curriculum. Finally e-commerce needs to be looked into and the ways in which students not only make themselves employable but self-employable.

Questionnaire Question 8.

What benefits if any do you see arising from gaining knowledge of computer coding?

Question eight revealed more positive contributions coding can bring on top of the more regular aspects discussed so far. There were some general statements such as wanting to learn something new and having greater understanding of technology.

“A base knowledge in any field can always prove useful.”

There were also some insightful comments such as the fact that universities could benefit from students with a strong web presence. Also students could increase their problem solving ability which is seen a core skill for computer programming. Designers who can code would be more generally clued up on technology meaning they would not have to reply on others as much. This could mean in terms of collaborative partners or needed people to act technical liaisons to facilitate collaborative discussions. It is acknowledged that coding is a growing industry and coding is a key skill increasing employability as it enables you to do work for yourself and for others, shaping the ever growing digital world. Computer coding is cited as being essential to the ever advancing technology in today's society.

The responses fall into two main categories: employability; and broadening the creative skill set. Students saw coding as a way to broaden their skills, open more doors and increase their successes in a career. Not only would the student CV be enhanced but more advanced job opportunities would open up with many possibilities to apply coding within them. Not only could it be used in the work created but in how the work is displayed or presented to colleagues internally or the clients in outside world. Self-employability is again mentioned as coding enables you to build your own business website.

The work created by the student with coding skills would then be more diverse, helping them to stand out in the creative industry and giving them the edge over competitors. Coding was seen by some as a whole new medium for creativity to thrive.

“Learning to code can give a whole new dimension to my work, as it gives me another platform to work with”

The idea that coding allows you to be free of software applications resurfaces with the concept of blank slate where you can create anything. This could stimulate creativities ideas that could not have been conceived until moving into this new medium creating the freedom to take your new ideas in unlimited directions. This stops creative resorting to shaping their work to fit an existing software or structure, limiting themselves inside the imagination of others.

Two interesting ideas arrive in the field of architecture that, in hindsight, have been mentioned before but have been ambiguous up until now. Firstly, programming intelligent buildings, or buildings with complex management systems to monitor carbon footprints, heating, lighting etc. are enabled through computer coding. BIM, or Building Information Management is currently big business in the world of Architecture and the appropriate knowledge at all levels could be advantageous to the architecture graduate, both in terms of how the model is presented (with possible interactive functionality) and how the building is actually constructed.

“the RIBA Plan of Works for Architects, the work stages by which a building progresses, has recently been overhauled to integrate BIM into it, which is an example of how computer coding and programs are driving the profession and making it evolve to embrace these new technologies”

Second, although it has been mentioned that coding provides an alternative to computer applications, coding also allows you to engage with some applications more such as when using action script in Adobe Flash or, GDL script in ArchiCAD. This further diversifies the applications of computer coding and a supports the need to create a diverse and flexible teaching and learning solution.

This interactive or intelligent element crosses over into the discipline of Fine Art where students may want to make interactive spaces and installations or even create ‘useful art’, a term currently in vogue in the fine are sphere. On-line galleries and interfaces to view art all over the globe are also growing which possibility could create more online curatorial practice. Generative art is also mentioned and Brendan Dawes is cited as an artist to watch who interestingly uses the Processing language as one of his tools.

Another interesting benefit is highlighted in that coding enables students to not have to rely on the use of prefabricated template websites such as WordPress and Cargo. These websites on the surface seem very useful. They are fully customisable and allow you your own domain name. It seems that some students would still prefer to have something that is totally theirs in design, construction, and ownership. This is admirable and further supports the argument for a diverse and flexible teaching and learning solution.

Ways in which people interact with websites, and the ways websites communicate to people are also changing. This includes mobile websites and apps. Companies with an advanced utilisation of coding can change the way visitors ‘experience’ the companies online presence. This has been seen in accompanying apps for shows and gallery exhibitions.

“Websites, for example, continue to develop new ways with which to communicate with their visitors through the symbiosis of code and aesthetics. A wealth of opportunities lie in being able to understand both the design of aesthetics and the code that builds a foundation behind it.”

The themes from question eight reiterate previous themes such as: Employability and creating the tools yourself; however new themes such as: enhancements to students portfolios; and universities benefiting from students online presence, both in terms of the work created and how the work is displays shows that the students are generally accepting of coding and the potential that it has to enhance work created by Visual Arts Students and the their school as a whole. Some students are happy to experience coding for the sake of it and see any new knowledge as beneficial. This is a very refreshing point to hear especially in the context of learning computer coding. Visitor experience is mentioned as well as intelligent buildings. Coding has the potential to create elaborate installations that me be abstract or functional that can be both the students own work and something for the institution to benefit from creating a strong relationship between the students and the space they inhabit. The benefits and individuality that comes from coding from scratch as opposed to prefab is mentioned, as well as the Software/coding hybrid model, which is the ability to create your own medium/tools/rules. This stops creatives shaping their work to fit an existing software or structure, limiting themselves inside the imagination of others. Finally the problem solving skills that come from learning coding are seen as universal and beneficial.

Questionnaire Question 9.

What problems do you think art and design students could experience when learning computer coding?

The answers to question nine are diverse and revealing and illustrated the key difficulties that students faced when learning computer coding.

Firstly, the actual complexity of it in an issue. The structured approach combined with what some thought was as an alienating language was seen as a major problem when learning coding. It was highlighted that this new way of learning was a regimented process and artists and designers are possibly not used to encountering these types of boundaries and hurdles. One interesting point was that it required a large amount of effort at the beginning for little reward. This amount of effort for the most basic of concepts was off putting and concentration was lacking. One student frankly stated that laziness can lead you to immediately say that it’s just too hard. The terminology or wording of the coding sequences, were seen as complicated which in turn made student feel intimidated.

Students were said to need a reason or to really enjoy the challenge if they were to engage.

"…as artists we tend to like the freedom to be creative and expressive. Although you can be creative with coding, I think coding in itself looks rigid, and has to follow certain structures and rules, which to many visual people may seem quite boring."

How the code combines to create functions was illusive to the art and design students. The technicality of it was confusing and quickly became overwhelming due to so many new and alienating concepts. Finally the debugging or problem fixing phase was time too time consuming and frustrating and they did not seem to be enough time to complete in time for deadlines. The tedious nature of the process left some feeling unstimulated. For some it is just not their field of expertise.

"Coding strikes me as a complicated 'art' in itself, of which I would not be sufficiently educated to enter at even a medium difficulty level, or have the sufficient basic skills required to even understand coding in its most primitive form."

Secondly, students felt that there was not enough time to get to a level where they could seriously engage. To get to a level where they can start to learn autonomously and start to make exciting projects of their own volition seemed too far away. One student highlighted the need for it to be taught at level four i.e. their first year in university. This makes total sense as they can start to apply their knowledge in subsequent years. This also seems to imply that if they came to university already with general coding skills, taught at A Level they would be in an even better position to engage with coding projects during their time at university.

Thirdly, one key discovery that surfaced was that students did not see the relevance of the technology within their field, or, more worryingly saw no practical application what so ever. As they don't have any knowledge of computer coding, they don't know what the technology does or doesn't offer them. This points towards a serious issue of awareness and contextual knowledge. One possible explanation for this is that students just do not have enough experience with computer coding and possibly not enough patience.

Fourthly, there are still incorrect preconceptions or stigmas attached to computer coding. Its seen by some as too complicated out of hand, too "nerdy", and too difficult to bother with at all. Also students think they need A Levels in Physics and Mathematics to truly understand. Coding can be seen as calculations and algorithms which it is but there are languages that are able to create building blocks through code “snippets”. Art and Design students may still find it hard to grasp and even restrictive but some of this boils down to a lack of confidence.

Fifthly, there is the notion that Art and Design attracts more people with conditions such as dyslexia which on the surface seems quite problematic in a language based medium. However the language and syntax i.e. the letters, numbers and punctuation; are quite mixed so this would have to be looked into further.

Many new ideas come from question nine. A possible hindrance to students is that there is just not enough time to learn computer coding. It is too complicated and too alienating. It can be quite daunting at first but surely there's no reason for it to be problematic. Some find it boring or not stimulating. Again, lack of interest, of the applications of computer coding is a recurring theme. Students need to be exposed to coding earlier. This may be being addressed by the secondary school system and the universities need to reposed to the students it is receiving. Some find their concentration to be lacking. Coding does require much time especially in the debugging process. Can visual or creative thinking keep up their motivation when problems arise and they have no formal training. Students need to develop more autonomous modes of learning. Self directed study will be important and this again requires a level of motivation that needs to be nurtured. Next is the concept of the creative mind. Does a creative mind struggle with more of the practical elements of coding in comparison to the creation of something visual without coding. Having a more creative mind rather than a computer/technology driven mind could cause some problems but I think they both can be adapted together to create a final project/result. An interesting avenue arises here which is that of dyslexia and what impact this has. How does a person with dyslexia respond to syntax as opposed to sentences. The stigmas coding has or the preconceptions that students have have may need to be addressed. Multiple ways of being introduced to coding need to be on offer to encourage engagement. Lack of confidence is another problem. Anybody can code if they're shown how. However there is an initial hurdle before it becomes accessible, and students need to understand that before they write themselves off. They need to be supported through this initial process.

5. Conclusions

Introduction

The research was original and rigourous. The questionnaires completed by the students and the interviews with professionals raised many ideas worth investigation. The methodology was well designed and the data gathering successful in that much informative and relevant data was obtained from both the questionnaires and the interviews. This was subsequently processed. The questions where appropriately worded and the ethical approach outlined in the research proposal was strictly adhered to. This makes the data relevant and of value. This final section draws conclusions from the information processed in the evaluation section of the research. Further literary sources will be drawn upon in support of these findings. It will first look at the main conclusions. It will then highlight the key findings, significance and ideas for further research.

First of all it is generally acknowledged that a shift is happening both in terms of what coding can be used for and how it is perceived by students outside of the computer science sphere. Previously most people, including most artists, looked on computer programming as dull and unimaginative, (Hickman 1991). Although Hickman found this to be the case he acknowledged that Artists are ideally suited to bringing creativity and imagination to software development. The stigma of I.T. or Information Technology is still apparent and is not totally accepted among these young artists and designers. The first thing that springs to mind would be the idea of the creative mind vs the logical mind. Research from the university of Utah (Nielsen, J. et al 2013) has shown that the left brained / right brained paradigm is not accurate and that people use all of their brain and can learn to think in many different ways. It is conditioning that makes people think that they cannot engage in particular subjects as they are not used to the thought processes involved. The so called creative mind can therefore struggle with more of the practical elements of coding to begin with, in comparison to the creation of something inherently visual without any coding involved.

There are still a number of perceived difficulties that may be preventing students from engaging. Programming is certainly a complicated skill to master, and learning to program is correspondingly complex (Jenkins 2002). Time is a constraint. Computer coding seems like a totally different subject. The learning curve seems too steep and the language is alien. Some cannot build the motivation to begin. Maybe they never will or maybe they lack the threshold knowledge. I define threshold knowledge as knowledge which, once attained, transforms the student's view of the subject, removing incorrect preconceptions of difficulty. This foundational knowledge may be critical in overcoming the apparent difficulty. How is it that coding 'creates' visual art. The threshold knowledge is what is missing that enables students to see the transition between the text and the visual piece. As stated in Hickman (1991) Students should be introduced to the graphical side on coding as quickly as possible and assignments should relate to the visual and conceptual possibilities offered by the computer. It is inconclusive whether some students lack the aptitude to program. Some research links programming experience with mathematical experience (Byrne and Lyons, 2001). Other research has stated that there is no way of predicting success with computer coding (Evans and Simkin 1989).

Awareness also seems to be an issue, both in terms of the capabilities and applications of computer coding, and how computer coding enables and transforms into more visual and physical artefacts. Finally, few students seem aware of the new and exciting incarnations of computer coding that may be far more appealing to visual artists, such as generative art and interactive spaces.

Different ideas relating to Teaching and Learning Theory have also arisen. Logical thinking is something that is required to write code and provides students with invaluable problem solving skills. How can this be generated with Visual Artists. Learning by accident or experimentation is also important which leans towards a more autonomous or self-directed mode of learning. How can this be enabled? Outcomes or Outcome Orientated Learning may have the potential to drive students forward in this endeavour and there seem to be good ways of stimulating students already apparent. Websites seem to be popular as does the use of interactive or intelligent spaces. Websites can extend into e-commerce through setting up online shops so the students can sell digital prints of their original artworks. The idea of creating your own digital tools has also come up. Computer coding created the programs we use to create digital art but sometimes you work within the confines of someone else's mind. Would creating your own tools be a more valid form of creative expression?

More universal ideas have also been mentioned, such as employability, collaboration and cross disciplinary activity. Computer coding is increasingly seen as a basic modern skill and students are acknowledging the fact that it could enhance their CV when applying for jobs in the creative sector. Cross disciplinary approaches are encouraged in the research sphere and evidence of collaboration is viewed favourably in the private sector. Computer coding arguably has the potential to expose students to all of these factors making a valuable learning experience.

In terms of learning, many interesting ideas relating to computer coding are mentioned. Firstly, outcome orientated learning seems to be relevant here. When students have an end goal in mind that they see as valuable, such as an impressive installation, a personal website or some other cutting edge application within their discipline, their engagement may be enhanced. Learning by accident and experimentation is enabled through coding. This develops the skills of the autonomous learner. Self-sufficiency seems to be a way to lessen the impact on the university and also to create ambitious students. Online communities offers a valuable support mechanism for the technologies that are not yet fully integrated into the curriculum. How can we assimilate these into the university support structure? Once students have the skills of not only the coder but of the autonomous learner they may begin to start building creative tools themselves. There is increasing amounts of literature within art and design research regarding the limitations that prefabricated software places on creative minds. Coding may be a way to break away from this. Paradoxically, coding also allows you to engage with some computer applications more than before. Certain design packages allow you to write your own scripts that can be integrated into the applications.

In terms of engagement, many factors need taking into consideration. Firstly there will be diverse methods of teaching. Some students don’t want to engage, some require structured module based teaching, some want after-hours sessions or clubs and some are completely self-sufficient.

What’s the best way forward? Due to the varying levels of engagement, students are discovering many ways to engage with coding, ranging from the simple to the more complex. You can code from scratch but also use prefab code. This is known as cookbook coding and is frowned upon in computer science circles. However to the artist this may be a valuable method to get results quickly while they sketch their ideas. Students want different outcomes from their programming endeavours. A learning environment that supports all these choices needs to be available. Allowing this flexibility has the potential to be inclusive of all students and enhance student satisfaction. Inclusivity is always important, and how we help students with different abilities and needs. Programming has to be integrated into the process and creation of interactive artefacts and not treated as a separate “"techie"’ programming module (Amiri, 2011).

In terms of visitor experience and organisational visibility, coding again can offer many positive bonuses to the institution. A student body that is more active online could increase a universities visibility. Coding can create a functional side to interactivity in terms of how buildings monitor themselves and change the environment appropriately. Student projects that interact directly with our own building would creative valuable press. Finally visitor experience can be enhanced through how they interact with university websites, how these websites provide additional support and information when they are visiting the building and the interactive installations or other visual stimuli they experience when in the public spaces of the building.

Main Conclusions

1. Awareness

Awareness of the technologies and libraries available and the applications of these technologies and libraries have been highlighted in both the questionnaires and interviews as key to creating student engagement in regards to computer coding. There is a perceived level of difficulty when students think about coding. This has been and is being combatted by the advent of new libraries such as OpenFrameworks and new programming environment such as Processing and Arduino. Before any coding can be delivered in a structured or unstructured way the works created by Visual artists both historical and contemporary need to be more visible and more common place in Higher Education. Students need access to media theorists and multimedia artists so they can harness data sets, realise their power as media manipulators and participate by building their own constructions (Mayo 2007). Coding is after all modern medium. It needs to take its place as just another material besides, fabric and acrylic paint. In a broad sense, the critical themes of art produced through computation are simply the critical themes of all art: the investigation of form and meaning. (Hertz 2009)

2. Collaboration

Collaboration is important and should be sought after. However, collaboration comes later. After something else has already been put in place. Collaborating effectively requires that experts develop understandings of one another's discipline as stated in Gooch (2005). Rather than expecting collaboration an early stage people from different disciplines could just be given a space to coexist in. This is happening already in establishments such as Madlab in Manchester and DoES Liverpool. Here people from different disciplines socially and work and collaboration happens more organically. This is a cross fertilisation process occurring rather than a prescriptive collaboration. Collaboration can also happen in the prescriptive sense. Brokering projects if managed well can bring together students who's roles are clearly defined and who's aims and objectives are well suited to the individual and the team. Although collaboration technically has the ability to lead to avenues that may not have been intended, this would enable the students to work together and complete something in the allotted timeframe that a module necessitates in the current H.E. structure.

3. The technician and the academic.

The support for this subject needs particular attention. The academic must be able to contextualise the technologies being taught. A critical issue in the use of digital technologies and digital culture within art and art history is the lack of context that it can bring (Cohen, K. et al 2007). They need to be active practitioners and aware of the new technologies coming out. The technician also has a large part to play in this inherently technical subject. Not only does the technician need to facilitate but also communicate with the various groups of students working with these technologies. This is even more important when working with students from multiple disciplines. Case studies have shown how the technical or professional communicator can act as a facilitator for people from diverse backgrounds who attempt to solve a problem and try to accomplish goals. (Marchwinski and Mandziuk's 2000).

4. Student Engagement

The research has revealed that achieving high student engagement is key to learning in this area and that different approaches are available to make this happen. Student involvement refers to the quantity and quality of the physical and psychological energy that students invest in the college experience (Astin 1999). A more collaborative and open form of education where students from different disciplines can meet and share ideas would create a greater potential for outcome orientated and project based activities. These have bean revealed as the best way to progress in this subject area. Pre-University level research has shown that the student's commitment of time and energy to academic work can be strongly influenced by student peers (Coleman, 1961;McDill & Rigsby, 1973). Through the research this has been shown to be a factor at the university level. However a dualism forms creating two separate pathways. These collaborative sessions could lead to innovation and startup companies being formed out of the university environment creating an incentive for some to engage. For others the creation of business will not be important. For creative students to engage successfully in their academic career they must have the opportunity to align their studies with their values (Reid and Solomonides 2007). Some groups may reject any involvement with computing (Hubbard and Grey, 1991). The most valuable pedagogic conditions, according to Reid and Solomonides, will be those that create learning opportunities that encourage this embodiment of the creative self. So it is important that creative coding can also be delivered in a way that is not indoctrinated into the maker community of business start ups and entrepreneurship. Understanding the complexity of creativity contributes to the nature and quality of student engagement.

There can be a long plateaux when engaging in coding for the first time. This can be lessened to combining coding with more accessible forms of digital technologies. A common and widely-used model of learning style is Flemming's (2001) Visual Auditory Kinaesthetic (VAK / VARK) model. According to this model, most people possess a dominant or preferred learning style. For example; The use of visual tools such as blogs and websites possibly with content management systems can ease students into the use of HTML and CSS and then further into JavaScript. Similarly a visual vector based or graphical starting point can ease students into the use of Processing. Physical tools such as the Arduino and Raspberry PI could ease students into the use of Wiring and Python. The integration of electronics or electronic art is an important one. Due of the uniqueness of these technologies in artistic expression, new modes of artistic communication and vehicles for artistic expression have been emerging. (Truckenbod 1988). I would argue that with all the recent advances in technology and online maker communities in this field that it's emergence is stronger than ever. Audio tools such as midi or OSC can ease the auditory learner and so on. This notion of easing is important as it gives the visual and kinaesthetic and auditory learners a break from the stresses of learning computer coding when they are from a non ICT background and would aid in keeping the student engaged. This would need to be supported with quality support from academic and technical staff to aid students to overcome hurdles. This approach would lessen the apparent difficulty which was a concern highlighted by the student questionnaires. The perceived complexity of coding by the students was inhibiting their engagement.

5. Constructionism

The notion of constructionism is a key factor in the use of coding in Visual Arts higher education. This notion has come to light through the interviews within the research. Constructionism asserts that people learn with particular effectiveness when they are engaged in constructing personally meaningful artefacts (Papert and Harel 1991). Seymour Papert defined constructionism in a proposal to the National Science Foundation entitled Constructionism: A New Opportunity for Elementary Science Education (Sabelli 2008) as follows: "The word constructionism is a mnemonic for two aspects of the theory of science education underlying this project. From constructivist theories of psychology we take a view of learning as a reconstruction rather than as a transmission of knowledge. Then we extend the idea of manipulative materials to the idea that learning is most effective when part of an activity the learner experiences as constructing is a meaningful product."

6. Possible applications

Generative Image or Algorithmic Art. This is the use of computer coding to create, change or otherwise enhance both vector and bitmap images. This correlates with the algorithms used in Digital SLR cameras to create jpegs through the internal picture profiles controlling factors such as contrast, sharpness and saturation.

Interactive Environments and Intelligent Spaces. This can be used for Fine Art Installations or Architectural "Smart" Environments, but can also be used to change or enhance Gallery and Exhibition Practice through creating more interactive and immersive experiences for members of the public.

Web Design or Online Curatorial Practice. The ability for Visual Artists to display their work online is important and is increasingly requiring knowledge of Web Coding (HTML5 (HTML, CSS, JavaScript)) in order to create modern websites that not only conform with modern standards but maintain the viewer's interest and are visually appealing.

Algorithms are now being developed that also affect moving image. Microsoft have released a new algorithm for example turning shaky handheld footage into smooth hyper-lapses. The ability for algorithms to drastically reduce workflow timescales will probably have a significant impact on how we process, view and experience the still and moving Digital Images of the future.

7. Further thoughts

Opportunities for extra curricular sessions or in an open studio environment or boot camps on coding can be made available for students of the the Visual Arts wishing to progress in this subject. This can be facilitated by technically minded staff and post graduate students. This would be driven by practical outcome orientated activities. Other technologies that interface with what the students are interested in would be made available. This can be facilitated by technically minded staff and post graduate students. Research has suggested that programming needs to move out of computer laboratories and into computer studios (Woodcock & Bartlett, 2005).

Students would also work or sit in teams who display their level of competency. This would cater for students at different levels of interest and engagement. The format would include a social element which may be lacking for some students in other areas of their academic career. Astin's (1985) theory of student involvement contends that students learn by being involved. Previous research (Krause 2005) repeatedly points to evidence of the critical role of peer engagement in the first year at university.

There is the opportunity to also bring together bootcamps from different schools for cross departmental engagement and collaboration. Collaboration however is not a term to be misused. It does not mean getting the computer science students to do the technical side of the art students project, it means working together and taking the project to places neither party ever intended it to go. Adding this diversity in the delivery would increase engagement by providing students different platforms and learning scenarios in which they could engage thus catering for different types of student. The inclusion of post graduate students would increase awareness of the capability of coding. Again, awareness was a factor highlighted in the research. Students from all subject fields in the Visual Arts need to increase their awareness if they are going to see ways coding can be incorporated into their field. These sessions would be voluntary and create a platform for open learning and discussion running along side the university learning programs.

This points towards a change in the way modules are created within higher education. As modules are validated their "Digital Potentials" could be identified. A framework of optional digital technologies could provide digital enhancements to each subject being discussed. These could be optional sections of the module or elective sections of modules, meaning that students could take different pathways within a module through the addition of a digital element. For example a product designer could either refine a product making process or physical process, for example the expert use of a piece of machinery, or they could use a digital process such as building functionality or interaction through the use of micro-controllers or micro-computers. Having these as two separate modules i.e. a machine process mobile and a digital module for example, creates the possibility of partial interaction by the class of students in both modules. By identifying the digital potentials the cohort can take a different path within the same module increasing the level of student engagement and creating a diverse student population who interact within the same space. This could extend beyond the use of coding. For example a visual artist dealing with the notion of time could deal with this in ways such as The Three Ages of Woman, 1905 by Gustav Klimt, of the works of Salvador Dali, or they could decide to take a digital pathway and use film editing in Adobe Premier or another time based software application. Again a supportive structure of competent academics and technicians and a variety of open sessions is nessassary to ensure success at a high level.

“The types of individuals who use and critically examine new technology will have an impact of the direction of cultural change. Art educators have the necessary preparation and orientation to explore the pedagogic roles of this new medium and to contribute their perspectives. I encourage them to do so in order to expand and improve computer application in our field.” (Ettinger, 1988, p.62)

5.3 Key Findings, significance, and further research

- From the case study of Liverpool John Moore’s University the students acceptance of coding is further on than previously anticipated. Coding is not only accepted but wanted and offers clear opportunities for visual artists and should be available and accessible to them. Universities across the country may need to move faster to cater for a growing demand across potentially all their degree programs. **How can we increase support for this growing trend while preparing ourselves for an influx of students which may have an exponential increase in understanding?**

- The way delivery happens needs to be fluid. Formal structures may be inhibiting growth. Cook book coding or borrowing and reusing code from others can be embraced to allow students to dip in and experience new technology while creating a fulfilling outcome. Different layers of support are needed to achieve this to cater for different students who want to learn different amounts of the subject matter. **How can we design courses that cater for in-depth learning as well as a project specific bite size form of learning, which also makes use of online learning, online communities and self directed learning?**

- Visual Artists and Art and Design Students may easily possess the potential and display an aptitude for computer coding and have the ability to engage. Teaching and Learning Structures need to be prepared for advanced use of these technologies. **What further research is needed to dispel the notion of creative minds and logical minds. Can we reconnect the technician and the academic or the technician and the creative. Are these ideas preventing the progression of people in this day and age?**

- Cross disciplinary collaboration may benefit from less formal meetings where people mix and observe rather than immediately working together. **Are universities structured to really allow true cross display collaboration or is it outsourcing masquerading as collaboration. True collaboration should take projects the places nether side anticipated it to go. Do degree programs have the time to accommodate this?**

- Self directed study is key to progress. Awareness is also key to motivate self directed learning. Students need to be shown work that not only exemplifies the possibilities but has relevance to the work they are interested in. This may include online learning communities but it may need more one to one support for artists who do not align themselves with social media and online communication. **How can we design courses that are inclusive of new ideas regarding online and global learning. How can universities adapt to survive in an online age?**

6. References

Amiri, F. 2011. Programming as Design: The Role of Programming in Interactive Media Curriculum in Art and Design [ONLINE] Available at: http://pers-www.wlv.ac.uk/~in5731/mymodules/docs/programming%20As%20Design.pdf [Accessed 01 January 15].

Astin, A. 1985. Achieving educational excellence: A critical assessment of priorities and practices in higher education. San Francisco: Jossey-Bass.

Astin, A. 1999. Student Involvement: A Developmental Theory for Higher Education. *Journal of College Student Development.* **40**(5) pp.518-529.

Banzi, M. 2009. Getting Started with Arduino. 1st ed. Sebastopol: O'Reilly Media.

Banzi, M. 2015. About. [ONLINE] Available at: http://www.massimobanzi.com/about. [Accessed 01 January 15].

Bloomberg, M. 2012. "Twitter Status Update." Retrieved 19 August, 2013, from https://twitter.com/MikeBloomberg/status/154999795159805952.

Bryman, A. (2008). Social research methods. Oxford, Oxford University Press.

Burg, J. 2013. Teaching Programming to Students of Digital Media and Arts. International Digital Media and Arts Association. **5**(1). pp. 23-34.

Burgess, R. G. (1990). In the field: an introduction to field research. London ; New York, Routledge.

Byrne, P. and Lyons, G. (2001) The Effect of Student Attributes on Success in Programming. *Proceedings of ITiCSE 2001*, pp 49-52.

Cellar-Jones, R. 2011. "Can the UK raise its game?". Retrieved 19 August, 2013, from http://www.bbc.co.uk/news/technology-15240207.

Chan Mow, I.T. 2008. Issues and difficulties in teaching novice computer programming. *Issues and Difficulties in Teaching Novice Computer Programming.* pp.199-204.

Chilcott, L. 2013. What most schools don't teach. www.code.org, www.youtube.com: 5:43.

Cohen, K et al. 1997. Digital Culture and the Practices of Art and Art History. *The Art Bulletin*, **79**(2), pp.187-216.

Coleman. J. S. (1961). The adolescent society. New York: The Free Press.

Douglas, T. 2011. "Google's Eric Schmidt criticises education in the UK." Retrieved 19 August, 2013, from http://www.bbc.co.uk/news/uk-14683133.

Evans, G. E. Simkin, M. G. 1989. What Best Predicts Computer Proficiency? *Communications of the ACM*, **32**(11) pp.1322-1327.

Fox, Z. 2013. "Learn to Code for Free With These 10 Online Resources." Retrieved August 19, 2013, from http://mashable.com/2013/03/13/learn-to-code-free/.

Flemming. N. D. 2001 Teaching and Learning Styles: VARK Strategies. Published by the author, Christchurch, New Zealand, 5th edition, 2001.

Gibb, A. 2010. New Media Art, Design, And The Arduino Microcontroller: A Malleable Tool. Masters thesis, Pratt Institute.

Gooch, J. 2005. The Dynamics and Challenges of Interdisciplinary Collaboration: A Case Study of “Cortical Depth of Bench” in Group Proposal Writing. *IEEE Transactions on Professional Communication*. **48**(2). pp.177-190.

Great Britain. Department for Culture Media and Sport. (2011). The Government's response to Next Gen. : transforming the UK into the world's leading talent hub for the video games and visual effects industries. London, Stationery Office.

Great Britain. Department of Education and Science. (2013). Consultation on the order for replacing the subject of ICT with computing: government response.

Greenberg, G. (1991). "A Creative Arts Approach to Computer Programming." Computers and the Humanities 25(5): 267-273.

Guzdial, M. 2005. Introduction to Computing and Programming In Python: A Multimedia Approach. Upper Saddlve River: Prentice-Hall.

Hertz, P. 2009. Art, Code, and the Engine of Change. *Art Journal.* **68**(1), pp.58-75.

Hickman, C. 1991. Why Artists Should Program. *Leonardo*, **24**(1), pp.49-51.

Hicks, J. 1993. Technology and Aesthetic Education: A Crucial Synthesis. *Art Education.* **46**(6), pp.42-47.

Hubbard, G. and Greh, D. 1991. Integrating Computing into Art Education: A Progress Report. *Art Education.* **44**(3), pp.18-24.

Jambalsuren, M., Cheng, Z. 2002. An interactive programming environment for enhancing learning performance. *Lecture Notes in Computer Science.* (2544), pp.201-212.

Jenkins, T. 2002. On the Difficulty of Learning to Program [ONLINE] Available at: http://www.psy.gla.ac.uk/~steve/localed/jenkins.html [Accessed 01 January 15].

Krause, K., 2005 Understanding and promoting student engagement in university learning communities. Keynote address at Hames Cook University Symposium, Queensland, 21/9/2005.

Kvale, S. (1996). InterViews : an introduction to qualitative research interviewing. Thousand Oaks ; London, Sage.

Law, K., et al (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education.* **55**(1) pp.218-228.

Levin, G. 2003. Essay for Creative Code. [ONLINE] Available at: http://www.flong.com/texts/essays/essay_creative_code/. [Accessed 01 January 15].

Levin, G. 2005. "Golan Levin." Retrieved 19 August, 2013, from http://web.princeton.edu/sites/slasharts/levin.html.

Levin, G. (2006). Computer Vision for artists and Designers: pedagogic tools and technologies for novice programmers. *AI & Society.* **20**(4). pp.462 - 482.

Levin, G., Lieberman, Z. 2004. In-Situ Speech Visualisation in Real-Time Interactive Installation and Performance. *Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering.* pp.7-14.

Levin, G. 2015. Biography. [ONLINE] Available at: http://www.flong.com/bio/en/. [Accessed 01 January 15].

Livingstone, I., et al. (2011). Next gen : transforming the UK into the world's leading talent hub for the video games and visual effects industries: a review. London, NESTA.

Mack, E. (2013). "Obama endorses required high school coding classes." Retrieved 19 August, 2013, from http://news.cnet.com/8301-17938_105-57569503-1/obama-endorses-required-high-school-coding-classes/.

Maney, K. 2014. Computer Programming Is a Dying Art. [ONLINE] Available at: http://www.newsweek.com/2014/06/06/computer-programming-dying-art-252618.html [Accessed 01 January 15].

Marchwinski, T. and Mandziuk, K. “The technical communicator’s role in initiating cross-functional teams,” Technical Communication, vol. 47, no. 1, pp. 67–76, 2000.

Mateas, M. “Procedural Literacy: Educating the New Media Practitioner,” On the Horizon 13, 2 (2005): 101-111.

Mayo, S. 2007. Implications for Art Education in the Third Millennium: Art Technology Integration. *Art Education.* **60**(3), pp.45-51.

McDill, E. L., & Rigsby, L. C. (1973). Structure and process in secondary schools: The academic impact of educational climates. Baltimore: Johns Hopkins University Press MIT Media Laboratory. 2003. Alumni. [ONLINE] Available at: http://acg.media.mit.edu/people/alumni.html. [Accessed 02 January 15].

MIT Media Laboratory Aesthetics + Computation Group. (2000). "Alumni." Retrieved 19 August, 2013, from http://acg.media.mit.edu/people/alumni.html

Neo, M., et al (2010). Designing a Constructivist-base learning environment: Using multimedia to engage students in a Malaysian classroom. In Z. Abas, I. Jung & J. Luca (Eds.), Proceedings of Global Learn 2010. (pp. 1496-1505).

Nielsen, J. et al. 2013. Evaluation of the Left-Brain vs. Right-Brain Hypothesis with Resting State Functional Connectivity Magnetic Resonance Imaging. *PLoS ONE*, **8**(8).

openFrameworks. 2015. About. [ONLINE] Available at: http://www.openframeworks.cc/about/. [Accessed 01 January 15].

Okamoto, M., et al. 2010. Computer Programming Course Materials for Self-Learning Novices. In: ed. World Conference on Educational Media and Technology, Jun 29, 2010 , Toronto, Canada. Publisher: Association for the Advancement of Computing in Education (AACE), pp. 2855-2861.

Orsini, L. (2013). "Why Programming Is The Core Skill Of The 21st Century." Retrieved 19 August, 2013, from http://readwrite.com/2013/05/31/programming-core-skill-21st-century.

Papert, S. & Harel, I. (1991). Situating Constructionism. Constructionism, Ablex Publishing Corporation: 193-206. Retrieved from http://www.papert.org/articles/SituatingConstructionism.html.

Partovi, H. (2013). "Leaders and trendsetters agree more students should learn to code." Retrieved 19 August, 2013, from http://www.code.org/.

PBSoffbook. 2013. Offbook season two. [ONLINE] Available at: http://www.pbs.org/arts/gallery/off-book-%7C-season-two/offbook-creative-coding/. [Accessed 01 January 15].

PBS Arts. Offbook. (2013). "The Art of Creative Coding." Retrieved 19 August, 2013, from http://www.pbs.org/arts/gallery/off-book-%7C-season-two/offbook-creative-coding/.

Peppler, K. 2010. Media Arts: Arts Education for a Digital Age. *Teachers College Record*, **112**(8), pp. 21182153.

Peppler, K. 2013. New Opportunities for Interest-Driven Arts Learning in a Digital Age. (Deliverable to the Wallace Foundation). Bloomington, Indiana: Indiana University.

Peppler, K. 2015. Dr. Peppler. [ONLINE] Available at: http://kpeppler.com/. [Accessed 01 January 15].

Peppler, K. A. and Kafai, Y. B. (2005). Creative Coding: Programming for Personal Expression.

Piteira, M., Costa, C. 2012. Computer programming and novice programmers. *Proceedings of the Workshop on Information Systems and Design of Communication.* pp. 51-53.

Popp, T. (2013). "Mooc U." Retrieved 19 August, 2013, from http://www.upenn.edu/

gazette/0313/feature4_1.html.

Princeton University. 2005. Golan Levin. [ONLINE] Available at: http://web.princeton.edu/sites/slasharts/levin.html. [Accessed 01 January 15].

Reas, C. 2006. Media Literacy: twenty-first century arts education. *AI & Society*, **20**(4), pp. 444–445.

Reas, C., Fry, B. 2007. Processing: a programming handbook for visual designers and artists. MIT Press.

Reas, C., Fry, B. 2010. Getting Started with Processing. 1st ed. Sebastopol: Maker Media, Inc.

Reid, A., Solomonides, I. (2007) Design students' experience of engagement and creativity, Art, Design & Communication in Higher Education, Vol.6, No.1, pp.27-39.

Ruttkay, Z. 2010. Programming for Artists with Processing. [ONLINE] http://archive.bridgesmathart.org/2010/bridges2010-555.pdf [Accessed 01 January 15].

Sabelli, N. (2008). Constructionism: A New Opportunity for Elementary Science Education. DRL Division of Research on Learning in Formal and Informal Settings, 193-206. Retrieved from http://nsf.gov/awardsearch/showAward.do?AwardNumber=8751190.

The Creators Project. 2015. The Creators Project. [ONLINE] Available at: http://thecreatorsproject.vice.com/show/zach-lieberman. [Accessed 01 January 15].

The Wallace Foundation. 2015. The Wallace Foundation. [ONLINE] Available at: http://www.wallacefoundation.org/learn-about-wallace/pages/default.aspx. [Accessed 01 January 15].

Truckenbrod, J. 1988. A New Language for Artistic Expression: The Electronic Arts Landscape. *Leonardo. Supplemental Issue* **1**, pp. 99-102.

Watson, C. et al. 2011. Learning Programming Languages through Corrective Feedback and Concept Visualisation. *Lecture Notes in Computer Science.* (7048), pp.11-20.

Weragama, D., Reye, J. (2012). Design of a Knowledge Base to Teach Programming. *Lecture Notes in Computer Science.* (7315), pp.600-602

Wikipedia contributors. 2015. Arduino. [ONLINE] Available at: http://en.wikipedia.org/wiki/Arduino. [Accessed 01 January 15].

Wikipedia contributors. 2015. C.E.B. Reas. [ONLINE] Available at: https://en.wikipedia.org/wiki/C.E.B._Reas. [Accessed 01 January 15].

Woodcock, A. and Bartlett, R. (2005) Software Authoring as Design Conversation. Proceedings of 17th Workshop of the Psychology of Programming Interest Group, Sussex University (online). Available at: http://www.ppig.org/papers/17th-woodcock.pdf (accessed 1st Jan 2015)

YesYesNo. 2015. About. [ONLINE] Available at: http://www.yesyesno.com/about [Accessed 01 January 15].

7. Appendix

7.1 Addendum

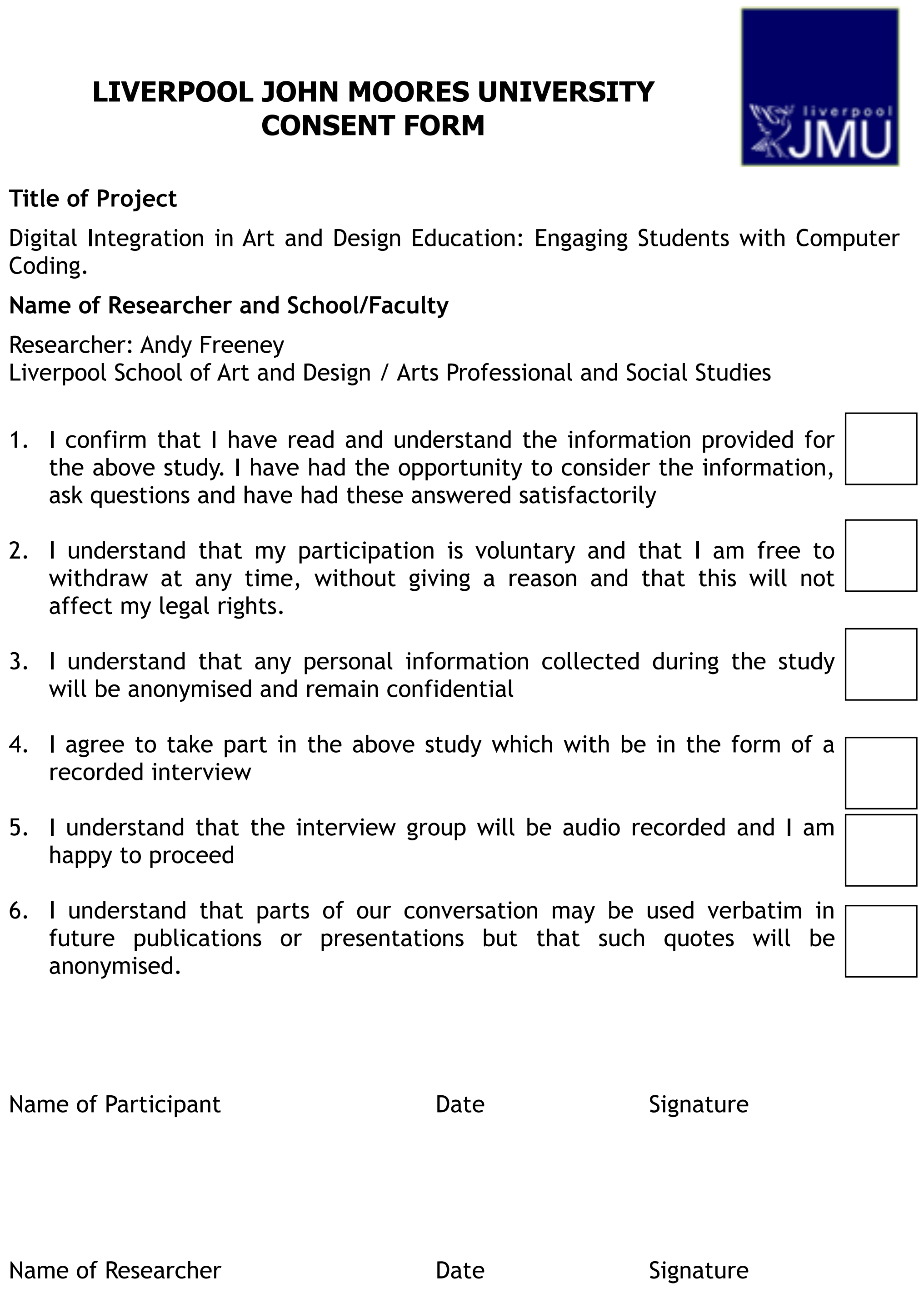
A potential transfer to PHD leading on from this MPhil is outlined here. A PHD project could be split into three sections. The first section would have comprised of the MPhil document submitted.

From this document the key points raised will form the basis of the second investigation. This would be action research into the points raised by the first section. This would be a reflective process on these key points which would have looked into it in depth. For example, student engagement is one of the key factors within the research. I would aim to research further into the modern theories behind student engagement and combine this with the cross disciplinary activity of computer coding within the Visual Arts. Outcome orientated learning and other existing methodologies would be combined within the context of a creative organisation looking to engage with modern programming languages. This section was not explicitly outlined within the original research proposal (rD9R) however it had been decided that this would be necessary to make sure the third section is built upon solid educational theory as per the research aims which are outlined in the rD9r.

The third section would be a practical application of the first two sections. A series of lessons possibly forming a module would be developed into key new technologies aimed at Visual Arts students possibly including technologies such as Arduino, Processing, Raspberry Pi, Open Frameworks and HTML5. Again this would build upon all the knowledge gained from the first two sections and would form the third and final section of the PHD document. It would have reference textbooks on creative coding directly relating to the use of computer coding within the visual arts.

There is a section in the rD9r relating to site visits to creative institutions. In order to obtain a broader understanding of creative coding in creative organisations links would be made with institutions outside of the northwest. where this research is based. These outside organisations included but where not limited to the Digital Culture Programme at Glasgow School of Art, the Faculty of Arts at the University of Brighton, and the California Institute of the Arts. In short it would be a foreword development of the practical application of the original research proposal.

7.2 Interview Forms and Information Sheets



LIVERPOOL JOHN MOORES UNIVERSITY
CONSENT FORM

Title of Project

Digital Integration in Art and Design Education: Engaging Students with Computer Coding.

Name of Researcher and School/Faculty

Researcher: Andy Freeney
Liverpool School of Art and Design / Arts Professional and Social Studies

1. I confirm that I have read and understand the information provided for the above study. I have had the opportunity to consider the information, ask questions and have had these answered satisfactorily □
2. I understand that my participation is voluntary and that I am free to withdraw at any time, without giving a reason and that this will not affect my legal rights. □
3. I understand that any personal information collected during the study will be anonymised and remain confidential □
4. I agree to take part in the above study which with be in the form of a recorded interview □
5. I understand that the interview group will be audio recorded and I am happy to proceed □
6. I understand that parts of our conversation may be used verbatim in future publications or presentations but that such quotes will be anonymised. □

Name of Participant Date Signature

Name of Researcher Date Signature

LIVERPOOL JOHN MOORES UNIVERSITY CREATIVE PROFESSIONAL INFORMATION SHEET

Title of Project

Digital Integration in Art and Design Education: Engaging Students with Computer Coding.

Name of Researcher and School/Faculty

Researcher: Andy Freeney
Liverpool School of Art and Design / Arts Professional and Social Studies

You are being invited to take part in this research, but it is important that you understand why the research is being done and what it involves before you decide to take part. Please take time to read the following information. There are contact details at the bottom should you wish to ask any questions or to clarify anything that is unclear to you. Take time to decide if you want to take part or not and feel free to ask for more information.

1. What is the purpose of the study?

The research project you are engaging in considers making computer coding a more fundamental tool for artists and designers in Higher Education. Computer coding is reaching a point in its evolution where it is entering the public domain and could arguably open up many new and exciting possibilities for Art and Design students even at the beginners' level of competency.

Through working with students over the past few years I have met those who, in their final year, want to create projects incorporating computer coding but find that they lack the knowledge or time to do this successfully. Not surprisingly they see 'Computer Programming' as another language with a steep learning curve. This is especially true if the student has zero or limited previous experience. Without the proper support one set back can result in the abandonment of ideas that have the potential to lead to exciting and inspiring projects.

I hope to correct this gap in knowledge by generating ways to help students to engage and succeed earlier during their time at university.

2. Do I have to take part?

No, you do not have to take part. This research is completely voluntary. Should you choose to take part you will be contacted to arrange an interview. Should you not wish to attend and interview an alternative online questionnaire will be available. You are still free to withdraw at any time. You will not be recorded as having withdrawn, or as having taken part as no personal date of any kind will be kept.

3. What will happen to me if I take part?

- You will be asked to attend an interview. This can take place in your place of work or you will be invited to the Liverpool School of Art and Design. This will be recorded and transcribed.
- Before conducting the interview you will be asked to sign a consent form.
- Should you not wish to attend and interview you can fill out an online questionnaire instead. You will be asked for your name and organisation.
- The questionnaire will take approximately 30mins to complete in depth but could be completed in less time.
- You do not have to answer every question.

4. Are there any risks / benefits involved?

There are no risks or benefits of this research other than you will have contributed to the research which aims to gain a better understanding of the field.

5. Will my taking part in the study be kept confidential?

Yes, any of your opinions used in the research will be anonymised. No personal data will be kept.

Name: Andy Freeney
Address: Liverpool School of Art and Design, Duckinfield St, Liverpool, L3 5RD
Email: a.freeney@ljmu.ac.uk
Phone: 01519041106

Name: Peter Appleton - Director of Studies
Address: Liverpool School of Art and Design, Duckinfield St, Liverpool, L3 5RD
Email: P.Appleton@ljmu.ac.uk
Phone: 01519041175

7.3 Interview Transcripts

Interviewee 1

The first point that comes out of the interview is the idea that people who are good at coding can work with people who are good at graphics and together build something professional. Through working in this way it is possible to advance into web environments such as content management systems. ***This technology introduces you to open source environments which not only applies to web but also to digital fabrication and interactive technologies.*** It is highlighted that although some people are capable of doing all aspects normally they focus on just one. This is however in the context of large in house projects for organisations and smaller projects may be possible by the individual. In larger projects there are teams working on databases and teams working on the front end or "skin" of the website. With environments like Drupal of Wordpress this backend section of the website is taken care of allowing people to work on the front end or "skin". ***However there is a cross over opportunity here by getting computer science students to create the back end of a website and art and design students creating the front end. This would mimic the workflow in industry creating a graduate skill set that would be appealing to employers.*** Those who cannot create a great design immediately can create static pages in photoshop as a single image that can be sliced up into a web format at a later date.

A lot of people don't see code as art and there could be a change in how it is perceived. There are movements in the UK promoting the fact that code is art. One example of this is the use of Processing. Even artists like "Beardy Man" are incorporating code into their music and works with a group of programmers to make BeardyTron which changes the way how he makes music. There is even talk of this becoming sentient. One of the reasons events like OGGCamp are run which is an open source festival is to promote that anyone can do coding and new open source environments like Python or Raspberry pi are making this more attainable and these are being sent to schools to start students off early with technologies that are easier to grasp. The change is happening in how technology and coding are perceived with new vibrant companies such as Facebook and google. Even artists are getting more into these new technologies.

Everyone should learn computer coding, even art and design students. It is part of everyday life and is engrained in everything so if you don't learn the language you a cutting yourself of. Too many people don't want to know how things work. They just want a box that you switch on. ***This has crossover with arguments against consumerism and how as consumers we are loosing the ability to think for ourselves and control our own future.*** We need people to be able to fix things when the break. Artists and Designers can engage in coding through technologies such as the Arduino. This links computer coding with changes in the real word or physical world. ***This has***

the potential of creating engagement by visual, auditory or kinaesthetic leaners who wish to see, hear and touch as they progress through the coding. This extra sensory feedback has the potential of keeping students engaged by creating an environment where play and fun are aspects of the learning process. The Arduino can be used to build art installations so art and design student should get not only into the Arduino but also those who use CAD and Architectural programs should have an understanding of how computers work. The blender project in Amsterdam is a 3D graphics tool. Its open source and was developed by a company who went bust. They realises the code open source and the online community has taken it and developed it to a point where it is very popular and even competing with programs such as 3D studio max. Even Hollywood movies are being built with this. ***This idea of the power of online communities is an important one. There is a whole online world that can assist students, encourage students and help students to engage in new technologies and discuss their ideas with likeminded students or professionals in online forums. This creates much opportunity for development.***

If coding was made more accessible students could engage easier. For art and design students making good looking websites using CSS and maybe JavaScript or jQuery would be a great route into computer coding. There are definitely jobs at the end of this because the web is always growing even if you only work on the front end.

However if people are generally artistic rather than being engineers they may have trouble in learning to think in the logical way that coding necessitates. People have different brains or personalities. ***Recent research has shown that the left brain / right brain model is a misinterpretation of a theory that did not intent for it to be applied so broadly. Humans use all of their brain and should be able to learn any skill. So what is the problem with coding?*** So you need think in a logical way like a computer. People think computers are intelligent but they are not. They have no powers of perception or the ability to change anything so you just have to lay things out for them logically and that can be challenging at first. People who love sodokus or rubix cubes should like computer coding because its all about logic problems. Most people like little puzzles so the problem might be just getting out of the artistic mindset and moving into an engineering one temporarily. ***This raises an interesting concept about the notion of self. People like others to perceive them in certain ways. If they model themselves as an artists sometimes they don't want to be seen associating computer science or with people in suits or capitalism for example. These ideas could be considered to be naïve or immature and leading a sheltered existence.*** Even though the interview has covered the idea that some people are better at art and design and some are more like engineers there are crossovers and everyone is a mixture of both types. Practice is important as it is with anything. If you are learning French for example it will be hard at first. Like when exercising you need to build up your muscles.

Q re: how to implement it in H.E.

Website creation would be a good way because its relevant to everyone particularly if you're an artist. If they had to do a bit of coding as part of this they may engage because they are getting something out of it. ***This idea of outputs is important. Students can learn by doing and be motivated in an outcome driven learning model. It can also be demonstrated that by knowing some coding even a little of CSS can increase the potential impact and individuality of a website making learning some code much more worthwhile.*** Student need to see the benefit from what they are learning. No one want to do it for the sake of it. So developing Wordpress themes, Drupal or tumblr can give them good skills and help them develop. This can be done before coding an entire site but coding an entire site is becoming an outdated concept and these tools are geared toward beginners or amateurs. Most people use Wordpress and tumblr. ***This idea has much potential for art and design students. They would not have to do the database coding which can be perceived as more difficult and instead design the front end or skin but still progress into more complex territory with javascript.*** In the "old days" you would code an entire site in notepad and this was good and gave you great skills and introduced you too all kinds of concepts but this may not be necessary. WordPress and Tumblr both allow you to migrate the site to your own website name or URL so they have two versions of themselves. The blog type environment but also the web hosting type. This means that students could continue their websites to be used in a professional environment.

Q re: Cater for different levels of skill

It can be very daunting when you first start of so its difficult to form a structure especially when some people are advanced and some are beginners. Project orientated activity may be the way forward. Using online environments for training and forums may be a good place to send beginners to get them up to speed. One of the Liverpool School of Art and Designers neighbours in the IC buildings are InterconnectIT who published a free guide for getting started with WordPress. This would be a great starting point for students getting into coding or doing something a bit more interesting. The benefit of this is that there will be more support outside of university online. These environments have large and supportive online communities. This includes raspberry pi, Arduino, WordPress etc.

Q re: How would the roles of the technician and the academic come into play.

Traditionally the two sides where separate and not enough respect between the two roles. There needs to be more crossover and collaboration. This used to exist in the open source world. Technicians need to be more interested in helping people who want to learn and academics need to appreciate technical ability. ***This also points towards a new form of technical academic***

someone with high technical ability but who engages in research and scholarly activity. In an online environment people can look down on beginners or “newbies” / “"noobs"” and can respond in an elitist or demeaning way to simple questions. This attitude is not helpful.

Qre: Inter disciplinary collaboration

Joint projects are the obvious place to start. You need to understand what other people do in other disciplines. If you work in a team with people of a different skill set its important to know what the can do some you know where the crossover point is. You could do job sways or experience another department for a week. ***Workshops could be run with students from different schools that could be held in different school or rotated around campuses.***

Qre: secondary school criticism. What about H.E

People arrive at uni and generally haven’t done any of this stuff unless they have been proactive themselves. Through looking into it the interviewee found that the curriculum was too simple and sometimes students know more than the teachers so when they arrive at university they don’t have the grounding in these technologies. The raspberry pi is pushing into this area and are trying to get the curriculum sorted out. ***This could lead to students with more intermediate knowledge entering university in all disciplines. Are we ready to receive these students.*** Coding used to be more in the curriculum and coding could be learned through creating games, however this all went away and people lost that ability and now other countries are overtaking us. The UK has a skills based economy and needs to be at the cutting edge of technology. People need to be more disciplines and to take these things seriously. The teaching is difficult because government policy is involved and sometimes this is steered in other directions. So school children are arriving at university without these skills and we need to make sure that we can compete with schools children from China for example. They are trying to fix the ICT curriculum and for the last 10 years it has been poor and now the education system is realising when degree programs from around the county receive these students who are lacking in ability.

Qre:Anything to add

Thankfully the world has changed. It didn’t used to be cool to do programming but now it is not seen like that. It is part of everything we do, smartphones, apps etc and people are wanting to know how to make apps. This may be another way in by making apps, frameworks exist that make cookie cutter type apps that people can then customise. This would be another good skill to take into industry.

Interviewee 2

how do you think computer coding is perceived within the UK does there need to be a change.

He comes from a fine art background. Graduating 6 years ago, studying time based fine art as opposed to pairing or sculpture. His introduction to coding came much later and came to it through Arduino. The Arduino was real game changer because it was so easy and there was such a network of support around it. It was also new and at he time didn't know many people who did it. Thats what he liked about it. Since then he's met many people that do it who he has collaborated with. Without that constant feedback and collaboration engagement would be difficult. He thinks most people require it. That's whats exciting about coding. At its source or heart it has open source and open access principles that guide it. That makes it very attractive as a platform to operate it. It is quite a nerdy thud but that can change if introduced earlier in education. At school level how do you prepare someone for a job when the nature of jobs is shifting. Coding is like learning a different way about thinking That way of thinking is the barrier point. Its different way of thinking. Learning how to create a process. Code and art is tricky because artists come from a lateral watt of thinking but coding requires a more structured approach because there is fundamentally input - process - output.

Theres not a stigma. I'm conferrable being a nerd. Technology has a divide. Digital and art, technology and art. You can;t separate it anymore. Its too integrated for you to have that divide. Young people can navigate an iPhone so easily. To them its so intuitive. A gestural interface is nothing new. Multitouch is nothing new. Coding is at the backbone of technology. The change can happen but its unsure how it will happen. If you look at silicon valley people can start multibillion dollar companies because people are in that sphere. There now getting coding into schools, at MIT they building software to enable kids to learn code through treating the structured approach. The process rather than the language. They can then use the language later. The fundamentals need to be embedded early on. When you start playing with an Arduino it doesn't matter what the language is. Its irrelevant. Its the background but someone else has written an interface that makes it accessible,. From a creative perspective the entry barrier isn;t as steep as it seems.

Its a very rewarding process because you can immediately start. Art and technology: Code allows you to interface the real world with the digital world. I can make a plant talk to the table, talk to the room and talk to the internet.

Technology moves at a mind blowing pace. Education can;t move at that speed. It will always be on the back foot. Whats exiting from an education perspective it that the idea of a text book can shift. The way in which we assess people can shift. We can tell how long it takes a child to read a page and turn a page. The teacher is not just being told that students got grades but the rate at which they progress and complete tests. All other bits of information are available to enable you to really measure a child's progress and their self directed study.

It seems strange not to have new technologies in the classrooms. The concept of time has changed now that people are constantly connected. iPads and phones are so intimate. There can be a problem with the technology having adverse effects. Like bullying that doesn't stop when children leave school or small children using I-Pads that are relatively huge in size and them being disappointed by the world around them.

Should art and design students learn coding and why

It would be great if it was an option. Its the same as electronics. Its such a hard job to allow his fine art department to sit in on engineering lectures. Its moved on from an object based environment. The outcome is not the main concern any more. The process is important. The university should provide a wide range of processes. Coding itself is art. A work of art that has the coding at its core has art within itself. If art is creating something then coding itself is art. You're creating something that acts in a certain way. It would be exiting for people coming into fine art and have this as an option. Once you are taught to think laterally you need skill sets to enact these ideas. Universities should provide a wider range of toolkits for artists. Coding is incredibly creative. Coding is art. It would be exiting for people coming into fine art education to experience coding. You don;t have to become the next heavy weight programmer. Its irresponsible for some fine art courses to teach they way they do because it puts them in a bubble and does not prepare them for the real world. Coding should definitely be an option. Not everyone should have to do it.

In what ways can we get students to engage.

Arduino wasn't the first micro-controller. Part of its success was iota open ideology at its core. All you have to do it post things on line. Forums have there own pitfalls but there are opportunities to share and post things online. Ownership has changed. Processing for artists or Arduino. It doesn't matter what the tool is. The ones that are more attractive are open access and grow exponentially. This idea of ownership in art is convoluted. There are producers coming in to complete different aspects of art work. With coding and digital art people are more that happy to share what they did. It would be pointless copying them. Artists should be integrated into society. Artists can be integral to change. These ideas are tanking along time to filter through. He is contacted to be asked how he built certain projects. Projects he could not have built had someone else not shared with him.

OPenframwokrds is a very powerful platform. The most impressive projects he sees are arudino , processing and Openframworks. Certain developments like open sound control from midi and dmx are interesting. Their not vey precise languages left over from a while ago but because their standardised they are very available or ubiquitous. DMX is standardised, but it is floored because the technology has moved on. Whats exciting about OSC is that it was designed as a way to interface devices with computers. it is incredibly powerful. 30:00

Different people are looking for different ways to connect things up and OISC is consistently being used across a whole spectrum.

Websites can be outdated in the the sense that they are not always updated and also how often do you return to websites. The good thing about websites is the fact that they are online communities. Communities of developers allow you to submit code to different projects and these submissions are judged by the community. Curation is also an interesting area. People can gather work together and put it under one banner.

Q WHAT PROBLEMS COULD STUDENTS HAVE

The teacher would be important. What happens when the student goes through the teachers lessons and comes back with a project that has already surpassed the level of the teacher. It it is going to be offered there needs to be scope for growth to a very high standard. Opportunity to visit other departments may be necessary if you are dealing with electricity on an engineering level for example. Projects were the technology is very visible is not as interesting when it looks like magic. The seamlessness of a good device is proof of excellence. If you are introducing people to these tools and technologies you need to be able to support them on a number of levels. You need to function across multiple departments. How easy is it for students to do that. Technology at its score is a very collaborative thing. There are so many other parameters that teaching in isolation will not fulfil. Using technologies in combination is more exiting. Coding can be lonely, you need to be of a certain mindset to sit there and work on it. When you are trying to do a project with no background in engineering and coding you can be lost when you encounter a problem. Encountering others who work in these areas such as DoES Liverpool or Madlab in Manchester has helped professionals in Liverpool in overcoming problems.

You need skills in asking questions online to illicit the right response. Also you need to be able to work through your own mistakes. Having someone correct something is not the best way to learn.

Q. How should computer program being injected into HE programs.

In his experience there have been lots of tools available but no real teaching. Rather than having a showcase it can be used on a project by project basis. Maybe they can use technology in a project.

they are already doing. Introducing it more subtly may be useful. Visiting artists could realise awareness. Workshops showing example, showcases of technologies that are relevant to the students work. The technology is integral but its not what is on show. Understanding input processing output is important and these can be substituted by any thing. Being able to understand at a core makes things easier to achieve. That would be a higher understanding than someone just playing with an Arduino. You can just pick it up and mash code together. You're not going to hurt yourself with an Arduino and some code. Unless you start to get into larger voltages. By more than one of your components because you’re going to break components. You need to come into the project with the motion that nothing is going to work and piece it together piece by piece. Being able to problem solve - that takes time to develop that thinking. Isolating bits of code or problems is a scientific way of thinking and an important skill. You don’t need to worry about how things work at a component level. Before you start teaching technology you need a counselling workshop to get you in the right frame of mind.

Q. What kind of structure would support different levels of proficiency.

This is important as a diverse student population is to be expected. Its important that the teachers are also practitioners as the use of technology is constantly evolving some people need to be onto of their field and actively engaged with it. When a student comes to you you need to be able to provide solutions to their problems. The best tutors were the ones actively engaged with the art world. Some times tutors will reel off the same list of artists to look at but none of them are current. These courses need to be modern. You need to be involved in the field so you can be aware of other courses around the UK or talks that would be of interest to students.

Q. How does the role of the academic and the technician come into play.

The role of the technicians in art is a complicated one. An artist has an idea but doesn’t always know how to realise the idea so technicians can be brought in to realise the project. The technicians role can be beneficial for the technician to advance their skills or even do research and development at someone else's expense. Completing projects to a deadline teaches you a different skill. Completing projects your contracted to do may make you achieve a higher standard. There is merit in doing private commercial work and doing a technician. As an artist he can understand what both he roles are. Ons of the most misused words is collaboration. Collaboration is an amazing thing, when the project can go in different directions to places you never thought it could. The term collaboration can be misused. Technicians should be ok not being a collaborator, they are a facilitator. Collaboration throws up a lot of questions about ownership and where the line is drawn. That relationship can be frayed and people can get disgruntled. Good relationships can be forged when both parties are working hard.

Q. How to we generate cross departmental or inter dis collaboration.

Massive questions come into play. How are departments structured, how are resources allocated, what time management system is in place. Other departments need to free time to allow for the collaboration. It looks very good on paper but its incredibly hard to implement. What is a collaboration. Is something going to lead. Is there an incentive. How would it be graded. Could any student in the team submit the work as a submission. Is the HE model conducive to this kind of agreement. Does this require a level of agreement from all the departments that isn't really there. I don;t think the sharing of resources is really there. Trying to get access to other departments was difficult in his experience. In his experience it was very difficult to get access to electronics classes. Artists will work differently from other department like engineering for example. If you've only ever worked with other artists you're coming to it quite blindly. Artists quite often aren't makers. They would benefit from being paired up with engineers. Can this be supported at an academic level.

Q. Secondary school HE coding.

How do you begin to teach kids. There is such a missed opportunity. The first country that teaches coding at primary level will make them market leaders. People need to understand how things work or their just package consumers. Different countries are building there own silicon valleys and they work because they are ecosystems. Its attractive for people in the world of computing. You could foster that behaviour in school. You don;t need expensive facilities for coding and electronics. It gets expensive when you move into big projects. You don;t need a new building to teach this. The personnel and the staff will be the biggest expense. How do you teach this? Textbooks aren't always the right way. There is merit in that way of learning. Project by project learning is the best way to learn. You not only need to train students to learn to code but also hoes to interact in online spaces. Education is shifting. It may not be shifting quickly enough. The internet provides us with a store of facts that we don;t necessarily need to know ourselves. We just need to make use of them. Differentiating information is still an important skill but itys how you use information thats important.

Q anything to add on the subject.

Interviewee 3

processing – flocking – style logo – Seymour paper MIT – reduced specially made languages. – Big Tracks –

Started working as a digital artist and coder and also doing youth work. From that entered education and working with digital media and young people. This informed art practice and collaboration with other artists.

Interested in the pedagogy of teaching and constructionist methodology and given people microworlds to explore with. Given people a small amount of commands and giving an exercise to work within. Logo and Scratch where designed around the same time. Something about the openness of these systems allowed people to explore new concepts and computational ideas. There weren't many artists working with this at the time. Seymour Paper cofounded the media lab and coined the term constructionism which is where the learner constructs their own learning. Some structure is needed. But only in terms of a scaffold acting as an educational framework that allows the learner to move freely within the subject matter and allowing people to collaborate with others through their own authorship rather than being spoon-fed. ***This creates an interesting avenue of investigation in creating an open environment where people can learn about new technologies, collaborate and explore new ideas.***

Q1 does there need to be a change

There has been a huge improvement especially now when organisations like FACT have a huge impact on the scene. Most people have an understanding of digital media art and FACT has informed this. If you look back 10 – 15 years there were separate camps of digital media art but that has become much more part of art practice and people are much more switched on across all art practices and see digital art as something to be integrated. It was nothing to do with art education 15 years ago and there are still a lot of art schools with no option in that direction unless you seek it out yourself. One positive thing from our current government is that they are trying to add computing to the curriculum but they are disconnecting creativity. It seems that what is being promoted in school sis that coding should be taught in isolation like you would teach French. There was a lot of great ground work down by organisations like *creative partnerships and curious minds* in finding ways for young people to use new media that was cross curricular. ***This is an important idea that creativity needs to be paramount when artists and designers are learning computer coding and cross curricular learning has many potential benefits .*** Young people could do projects that involve history creative writing and programming rather than just learning programming in isolation. It needs to be meaningful. Making an app is great but young people need to make something with a purpose and seeing as technology is pervasive its important that it has a pervasive role in the curriculum instead of being pigeon holed into one subject area. ***Seeing technology as being pervasive gives insight into how it can be in included in multiple subject areas by perceiving it in this way.*** Creative partnerships where set up by the labour government and came out of a presentation by Ken Robinson. They still do loads of great work in schools.

Q2 should art design student lean CC

Absolutely. At the very least it gives them grounding in contemporary culture and where technology is going. Students should have an understanding of computation and how it works. If they're not coding directly they will probably collaborate with people who may be building more complex systems or building systems for them so you need a basic understanding to engage with that and also to engage culturally in aspects such as open data, open source culture, and the effects of creative people are organising themselves and distributing their works and huge effects that has on culture is huge. ***This creates a very good point that student may require the servies of people with skills in these areas if not doing it themselves so a basic knowledge is important either way so as to get the best deal and the best representation of your needs.*** So students need to be aware of that in more than a detached way. They main reason if for its creative possibilities. Coding and programming can open up different ways of thinking about things. This happens to be the most powerful tool we happen to have now – computation – when we look back at van gogh, rembrandt, wright brothers – they would be playing around with this stuff now. In a way the write brothers were media artists working with their new toolkit.

This creates a strong argument that the artist is the person who works with the newest tools.

To flip the argument around why would anyone make the argument to only use traditional media for art students. A lot of the most interesting ideas are around authorship, the hive mind, what is changing in society, require interactive, or require that art works not be bounded things. There was a period up to post modernism or you could argue up to modernism where artists produced objects of contemplation, but since the avant garde artists of the 20th century and media art there is the idea that the art object has been decentralised and art practice and process become more meshed in the social fabric and social construct they are part of. And programming and computation is an important part of how that society is working now. So it's not just an object but something that manifests in the world. ***This Discussion of art history and computer coding place in it creates a new avenue for investigation and another possible way of getting artists to engage in the technology b showing its place in the art history timeline.*** If artists want to create works that aren't bound by time or space, like galleries etc then computation is an interesting way of doing that. Another big sell is that You can also make art projects that are interactive or that do things.

Additional Q – hearing you mention all this makes me feel that I'm asking questions that have already been poised and is this research just a rehash of old ideas.

Vercoscky and the constructionist guys wrote there stuff over a hundred years ago and there ideas still aren't probably integrated so these questions are still relevant . How old an idea doesn't really matter. If a method is good you should continue to make the argument for it. The student who comes to an art school now are different to student who came to art school 10 years ago because they are digital natives who have grown up with technology. Also the technological capabilities have totally changed. ***This raises very interesting points regarding the digital native and how these questions should be continued to be asked regarding how to integrate these technologies which have all moved forward and being engaged with by students who are increasingly more tech savvy. Add coding into this mix and you have great potential for innovation.*** Older multimedia courses where just getting excited about having things move around on the screens or programming screen based interactions and now people are exploring these internet of things or maker based interactions and this still hasn't reached fruition in a lot of art schools and it feels like you have DoES Liverpool who are experts in the internet of things and world leaders in that so it seems quite ripe for an art school like Liverpool john moores to start taking on those ideas too. ***This creative interesting argument for outreach and external engagement within the city of Liverpool.*** However all this being said coding should just be spread really thinly across the curriculum as opposed to being taught. It needs both really. ***This informs how to integrate coding into the curriculum. It would need some classes and also opportunities to embed activities within modules.***

Q – How to integrate in degree programs.

An intro to programming classes would be useful using something like processing or style log0 – or other entry level programming languages. Most students should have the opportunity to do afternoon workshops or classes, to get to grips with simple ideas in programming if – then etc. Or getting things to animate or move on loops so that people can get an understanding in order to break down some of the myths of what programming is. All students should get an induction using really accessible programming environments and maybe to situate that is a half day or day challenge to make a simple game or app or something. Processing in particular has got the capacity to prototype really quickly and move it onto android so that is an area with much potential for students. ***These day long sessions though interesting could have a great impact on the school timetable so holiday weeks, quiet weeks or summer schools could be set up to cater for these. This has the potential to mix students with members of the public creating opportunities for external engagement and the similarities external engagement has with open source movements and online communities.*** The main emphesis of these days should be for students to get to grips with the technology but also to make the connection between coding and creative thinking and making creative things. Also coding lessons or workshops should be part of the curriculum of the degree program. Maybe 2 hours per week could be some coding or multimedia activity to boost the links between the art students and outside bodies like the maker community. It would make sense to make programming part of collaborative design modules. Whether it's a module for making a prototyping device or clothing. Etc and you had to make it interactive you could then run introduction to Arduino workshops or lilly pad for example. And build it into those iterative design cycles. These have been adopted a lot by art schools are give students tried and tested pathways. Evaluate , design test etc. This is used to make interactive prototypes for things. SO you can include programming in teaching or digital computational teaching within some kind of challenge or product design or art work where students create versions of things, evaluate things and prove them. You can use that framework and then narrow it to suite a particular technology that you are looking to share like a wearable or interactive clothing or some Internet of Things product.

You could provide educational support for them doing all the user research, design methods etc and also technical support. ***This shows the best way for academics and technicians to work in unison covering all aspects of the discipline.*** In terms of how it should be taught constructionist teaching methods would be best using challenges and collaborative exercises with all the tools laid out more like kindergartens.

Q – How to get student to engage

You should have a scaffold with enough resources to support them to not too restrictive. And then tutorials on computer language such as processing. But to emphasise self directed study. ***Self Directed study is an important concept to engrain into students***

Q- what problems could art school students experience.

One of the most difficult things is understanding fundamental concepts of computational logic. You need to see the big picture as well as the coding terms. Art schools attract visual thinkers and coding is very different to what they might have thrived on in the past. That can be off putting. A barrier could be learning that you spend most of your time testing things with things not working so getting to grips with debugging as being a creative process, so there you do need face to face time with support so people don't give up when they hit a wall and think they can't do it. ***This highlights an important part of the learning process*** Patience and persistence is required. Critical thinking is also important in order to move away from a trial and error mentality. ***CRITICAL Thinking is a key skill to be developed in HE***

Q- how fed into programs – mentioned but already covered.

Q- what structure support different levels of proficiency.

A mixture of formal teaching. The more open ended the activities are the better. If they are being taught in a way where the student are given a certain amount of code to explore with a number of different avenues then they can fulfil their interests. They can make something simple or more complicated. As they get used to playing with it they can be assessed by what they can do with it rather than everyone making the same thing and completing a set exercise.

Setting up informal spaces where students can work in an open way like a hack space where there is lots of peer support and play. ***PEER SUPPORTED LEARNING IS A KEY Concept***

People can be separated into different groups of confidence rather than mixing them up because one person can race off and do everything. New comers can then be brought up to speed on a one to one basis and then added to a group. Also having online content or lecture notes available prior to the session can also be useful. Prepping students can support students of all levels and online resources can give people the confidence to engage. Even a blog or a wiki that people can check and investigate at their own pace. This is important.

Q – roles of academic and technician

Both have roles to play as it is quite hands on teaching. Supporting students explore these ideas. The academic could have more of a research or contextualise what is being learned. Goldsmiths have a computational art department within a computer science department which is interesting and generating really interesting practice. Art practice as well as technology practice was happening. NB – GOSSIP ***In Newcastle they have a media arts masters in the fine art school but the computer science department creates much interesting inputs.***

Having access to real techies in computer science is really powerful but there is a risk of communication going astray when you think you're going in the same direction and then a schism forms.

At fact when the tough izadora there was a general understanding that they were learning technical stuff but for artistic purposes. A lot of that was to do with the tone of the delivery ***This is a very interesting avenue about people motives , delivery and end goals people may have a different agenda people need to be on the same team***

*ben fry – organic design free

Q – promote interdisciplinary collaboration

Brokering – don’t let people take too much ownership. Invite schools over. Students could work together from cross school with geographers and history students. Creating meaningful projects. Not just art and computer students. Having a public realm output could really motivate students – to be publicly engaged. Having students deign something for a group of people and collaborate with those people making something in a particular context. Having a reason other than just to pass the module is important. That’s the difference between an art practitioner or professional artist and a art student is that the steaks are higher. If you have public realm engagement and output it forces you to reach a level of professionalism. Otherwise group projects can make you feel like you’re jumping through hoops without at proper output.

Q – uk secondary criticism.

Art colleges might not be ready for a new influx of computer programmers. There is a lot being talked about regarding this generation of digital natives. So universities may get more digitally literate but not everyone will learn to code. This idea that we will receive a tidal wave is probably pipe dreams.

Q Anything else you would like to add.

Interviewee 4

How is computer coding perceived in UK and in art and design does there need to be a change

It is difficult to perceive being embedded within the coding community. The is a movement to teach kids to code, and it is said that today that everyone needs to code as opposed to just learning Microsoft word or typing. This is an important skill to learn. The preview model taught you how to use office software but did not teach you anything about a computer. There should also be an opportunity to learn how to code. Douglas Ruscoff's program or be programmed lists ways in which programming can be a useful skill and applicable in all areas of life and also is relevant in the use of software applications which also for user input in the form of code. This is more general coding rather than Art and Design, but if this culture starts in secondary school this will affect the students entering H.E. Adrian comes from an engineering coding background rather then the arts. Since being in Liverpool at times he is seen as an artist. Collaborating with FACT there is a lot of digital, arts going on. Digital Art is not always that good or look that create or coded very well. A friend of his that went to school was very musical. He could not play any instruments but he had worked out how to use software. He then started writing code to generate samples. He would write programs to spit out steams of numbers that made noises when played as a sound file. He wasn't that cutting edge but for his stage of development he was pushing his limits. He was using the tools he had a s a computer science degree.

Q Should Artists and designers learn to code.

If they're going to do digital arts then they should. There is always a tension in that on one level you want it to be accessible to everyone so you have an amateurish as in not formally trained situation of people getting into building websites for example. But at the same time there comes a point where you would have coded something differently and better and more robust if you had a better understanding of programming. Things would be scalable for example. There is more opportunity for people with a computer science degree so it would be better for people to no how to code properly but it should halo be available to people wanting to try it out.

Q How do we get art and design students to engage in coding.

Part of the reason the Arduino came out was to for the interaction design students to learn coding and electronics. It was made for artists and design but established at the Interaction Design Institute. (IVREA) It was made to teach the design students there who were very much graphical, product students and none "techie" with no formal coding tuition. This tool made it easy for them to pick up the fundamentals. These tools are useful by showing people the possibilities while making it easy to make breakthroughs in their development quickly. The Arduino and a few lines of code can create a more interesting experience when dealing with product mockups or presenting work in degree shows. The project orientated approach makes it easier to learn. Problem solving is better for learning technologies. However an understanding of teaching is necessary and there may be other approaches.

What problems do you envisage art and design students experience when learning computer coding

As a practitioner rather than an educator I don't have that many tools in my toolbox to approach learning ion different ways. It feels at times that there are different people who have different ways of leaning about things. When attempting to paint he realised that the artist can paint things with different colours based on the interaction with different objects in its environment whereas he would normally paint things with the colour that the object is. Similarly the artist may struggle interpreting an array and how it works within computer coding. To understand what an array is in not complicated. But they array can serve several different purposes. But they are all just mathematical values. They're inherently none visual. This may be the stubble that visual leaders are having. The best way to find the problems that the students are facing would be to talk to them. However there are tools such as scratch, which is visual, also If This Then That allows you to layout blocks to understand logic. HTML is useful as it sits between programming and the visual. Its functional and declarative. This may be a way for people to start and get their head around coding.

How could computer coding be fed into art and sign programs

Getting people more into using the web and exposing them to that. People don;t get forced to build their there own website. Not necessarily from scratch but having a website or blog is easier enough these days. Its useful for people to be online. This enables people to have online portfolios and it can make them more employable. People with blogs can show that they are engaged with subject matter and write about topics that may be of interest to employers. This creates a gateway into coding through customising blogs and opens the doors to the power of what is inside computers and the ease with which it can be accessed. In computer science degrees all types of coding applications are covered to ensure that the students are exposed to different topics. Exposure is important and also forces students to engage on some level. Small credit modules may be a way to do this. It is not for everyone so this could be ban elective. Possibly a module on generative art. SO it has a basis in art but includes elements of coding. Students who did not think they good be good at it may tun out good at it or quite interested in it. Using Processing to make generative art may be successful and may expose students to the cool things that can be created. To the artist it can be treated as another material. Students will react in different ways. Some may think it was difficult but its easier, some may think it's still difficult but it can create amazing things or the things that I want to make. You could do a module which is very graphical and computer art and then also one which is physical involving the arduino. This could be analogous to oil painting vs sculpture. You could steer away from the fact it digital and computing and just treat it as art. A behaviours module could come into product design to make models interactive or behave in certain way. **Coding is just another material. There is a lot of frustration but you need to get involved and get your hands dirty with coding**.

Q What kind of structure would support students of differing levels of ability.

It depends. There are always issues of staffing levels and resourcing. In a perfect world you would have people who know more about all aspects. For some they are able to expose themselves to different things with support. There are able to go away and think about things and come back and make progress. Its hard to assess that situation. You have different levels of proficiently in both the students and the academics. **Peer supported learning may be an option.** This could be tutorial groups were you have lecturers who know their stuff mixed with post grads who no more than the undergrads who could possibly run some tutorials. That could help with some of the staffing issues. Post grads could answer questions of the beginners and they could use lecturers to answer more complex questions. Schools use different levels or sets for different subjects. A constructionist method is however employed in that tools are provided for the students to explore and support is given when necessary.

Q The technicians and the academics.

There are two types of computer technicians one type is the traditional university technician who fixes computers when they aren't working. This comes back to the idea of having to get your hands dirty. There are technicians who will get their hands dirty when the lecturers will not. It seems like technicians are more hands-on with equipment. This is just his perception of the technicians he has experienced today. Often the technicians won't get the credit they deserve, some technicians do just give access to equipment and there are others which are more interested int heir discipline. Then they are getting more towards being lecturers but that is an artificial distinction. A lecturer Tom Igoe who is a lecturer in new york. What they try to do is have… The stuff they teach is a post grad course with all kinds of digital techniques to build interactive art works. Its covers art and design but has a strong digital strand throughout. Its a bit like Maker Space. Mr Igoe was not sure what the role would become in five or ten years. They are building innovative stuff but its more about the tools that they are using. Its leading edge tech but leading edge commercial tech rather than scientific. This is a new way of working, rather than 3 years at a university they will have 3 years at these Maker Spaces, sharing things on the internet and learning online. This may not spell the end of universities due to high end research. People could create their own learning instead of being exposed to modules they have no interest in.

Q How do we generate cross departmental or interdisciplinary collaboration.

He has met many people from around the university. Where he works they generate lots of disciplinary coloration from the fact that the people are there in the same room and are from many different backgrounds. This makes cross over and cross pollution possible through general conversation rather than any structured approach. People who contact the establishment have those kinds of interests and they meet people from the university who are interested in this topic. There should be ways for crossover not just between the arts and commuting but also with business and entrepreneurship to enable startup organisations form. This doesn't seem to happen because the student don;t get to know each other. He is not sure whether the issues are economic or political. Maybe the lecturers or the people in power do not see the benefit of interdisciplinary activity. Its possible that LJMU has more of an interest in academia instead of creating people for the private sector. The university should not be to create a conveyor belt of professors. **University should prepare you for work and creating businesses.** Research is important but it should not be the only option as this will create a disconnect with some students. **Teaching how to learn so you can learn other things is important. Teaching a language in a way that lets you learn other languages would be beneficial.**

There is no need to resign the university from the top down but if space could be created for different approaches then this would be beneficial.

Q Secondary school criticism.

Its good that we are getting more programming in the curriculum. We need to be ready to receive those students. Compulsory programming will be the question. If it is compulsory then that could be good because artists and designers would have experience however they may not elect to do it. The university may not necessary receive more interested people but those who are interested in coding and the arts may be more skilled up. This may be beneficial to a university looking to support these activities.

Q Anything else you would like to add.

Interviewee 5

Question 1

Coding is broadly embraced and most people would want to see its use in Art and design education – more prevalent than it is. No one would say there wasn't a place for that today. They used to say that but times have changed. Code as an aesthetic and easthetical form has much discourse around it. ***Code as an ethical form is a good view from an art perspective*** It is embraced but not an easy one to embed an instil in students creative palette, what they could be doing in an art school. And for a number of reasons. There are examples of how it could work and how it could work better. You could send students to a computer science department. This may be successful for one or two however it will not work, I suspect for everyone. How successful that is I don't know. There are different pedagogic systems in place in different schools and different ways of acquiring skills. ***What different pedagogical systems exist ion art schools and computer science departments.*** The environment of computer science schools are different to the studio environment, this is what students are used to and changing this may not be conducive to their practice and their learning. There is a different learning methodology involved. A lot of students may be surprised by being faced with programming when leaving A-LEVELS. Students may be taken aback by that. Students friends and family might also be taken aback. So this is also a challenge – how you deal with students preconceived ideas of what they are expecting from university – This would be quite a challenge. **This points towards the option of an elective or at leasrt an alternative to computer coding.**

Questions2

The principles of coding, computing and networking are important to learn for a student. **This points towards a General Computing Knowledge book for the laymen to understanding.** Not about any particular language – but knowing what it is and what the potentials are of it. If they can grasp what coding is and what computing is they would be much further on in their development. Its one thing knowing that it is conceivable to get there. If they understand the way to get to their goals it would be better that expecting that it is magically going to happen. Collaboration is important with learning. Some students will be better getting their head around different packages. Some will be more mathematical, some will be more object orientated and some may be more physical even i.e. Playing with hardware and hacking. ***What different skills or approaches are there to computer coding*** . Coding isn't just about sitting infront of a computer and crunching algorithms it is about systems and if you can think of ways to make systems arenot just involving digital then you are introducing students to it far better. Give them a feel for creating complex system. Max / MSP or quartz composer is an object orientated way of learning involving plugging systems together – like a breadboard – experimenting – playing around and enjoying unintended or unanticipated outcomes.

Questions – engagement

Focus development around outcomes and projects ***OUTCOME ORIENTATED*** - You can offer any kind of coding – coding written down on paper , coding that is sculptural , coding that is numbers and mathematics but you ultimately have to describe the outcome that can be discussed using the same methodology across these different outcomes. A project was developed using alan turing because it was his centenary. ***USING REAL WORLD EXAMPLES THAT ARE HISTORICAL – CROSS DISCIPLINARY – MORE INTERESTED TO LEARN*** this WAS something they could grasp as a subject to deal with. If you try and break int down into systems and different ways of looking at them. Students can be very good at different things – like skills in processing – when studetns see a skill it is not something that they become tassit about – this is something that they share and they can weave skills together and share there skills – This is why graduate teaching programs are good. **Bring back graduates to do sessions with student** Everyone has a skill. When people meet each other and share skills over a coffee they get exited about what other people can do. **What are my skills – what am I good at how can I contribute.** That's how you communicate ideas. Not by setting everyone the same task and seeing who does it correctly. **Spoon feeding and teaching by numbers has got to go how to we create the dynamic for real development and learning. Maybe coding is just a tool for collaborative learning** Create a dynamic where they can all help eachother.

Question 4 what problems

The athitheses of what has been said. The assumption that its going to be learned like in a maths class. That there is only one way top learn. You need to get over that. That's the biggest hurdle to get over. Once you are over that – that has nothing to do with what you are now teaching and studying. You need to identify an outcome but not be totally fixed on what it is going to be **Allow rroom for manoeuver when making outcomes** find the way to get there – there will be some getting to grips with demos. There will be failure and problems but what ever you come out with must be useful to you. Its ok to copy and paste and use other peoples bits of code and hacking other peoples stuff. Rather than saying here are the lessons - now code it from scratch. The more sophisticated you become to possible methods the more you can connect with online communities and discuss. **Online communities.** You need to find these commuinities. Everyones been whaere you are now. There is lots of documentation of other peoples attempts and mistakes. When you look at a complex computer game this ionvolved large groups of people all working on different aspects. The more suceeful people will show you they way – show you the ropes. Yo need to immerse yourself. – Students will need to give it time. **GIVE IT TIME ITS ESSENTIAL** Its going to take some time and some evenings – throw yourself at it. Find a problem that you want to solve. **PROBLEM BASED LEARNING _ SELF DIRECTED PROJECTS** spend time in contemplation of these problems. Some students will want to get there as quickly as possible and will want to move quickly – this may not be possible. And they may not be prepared to spend 6 weeks on something. **This may not be for everyone even those who think they want to do it.** Those who spend the time will discover much more that they expected. They will need technical help something.

Q5 how to feed it into projects – what are the best methods

Technology must be embraced. Coding is not for everyone. Some people could be steered away from what they want to do and marginalised. It has to be available in a way that - It must be provided for on one hand by the IT division and on the other hand in the lab environment – drop in – install your own software – bring in your own hardware – plug in – take it apart. – plug in Arduino – download processing – open environment. So you need two environments – one is the traditional computer suite and one is an open 24 hour environment thas is all kinds of technology connected to projectors – sound devices etc. **This points towards an interesting idea that digital culture can be taught in the standard compouter suites and connected to the open environment – so students learn holistically and both environments are discussed in the context of each other.**

POINT B We need an open Lab Environment.

There are multiple ideas here – A programming environment – a hacking environment how to we link these.

Q6 What support structure for different abilities.

It needs to be research project driven. It needs to be a centre where things are happening and you are building a community. Working together - Apply for commission – apply for more kit – bring students back into that – successful students who have adapted will help – they are familiar with the territory – they may want to do a bit of teaching – do an MA – connect with industry – connect it all together – you want to avoid the situation where it is hassle for someone to maintain the space – it should maintain itself. You can build it with live projects – live briefs with industry. Having stuff going on – promotion documentation. – **HOW TO BUILD A COMMUNITY** Have links to outside – like madlab – fact. Links to outside populates the community – students staying on in the area starting as an artist.

Q& roles of tech and academic.

Roll of academic it to keep up to speed with the students and to know all the things can do. Know from experience what is good to work with and what other artists have done. Contextualisation. **Raphael lazarno hemma.** Who artists work with – who assists them – what technologies they use . know the context they are working within . Have a grasp of current technologies. Know - not your limits – but what is worth knowing and what isn't – you cant know it all. Know when to stop on a different path. **ROLL YOUR OWN _ CODING – talk BLAST THEORY – KenGOLDBERG**

Q Cross disiplianry cross departmental collaboration

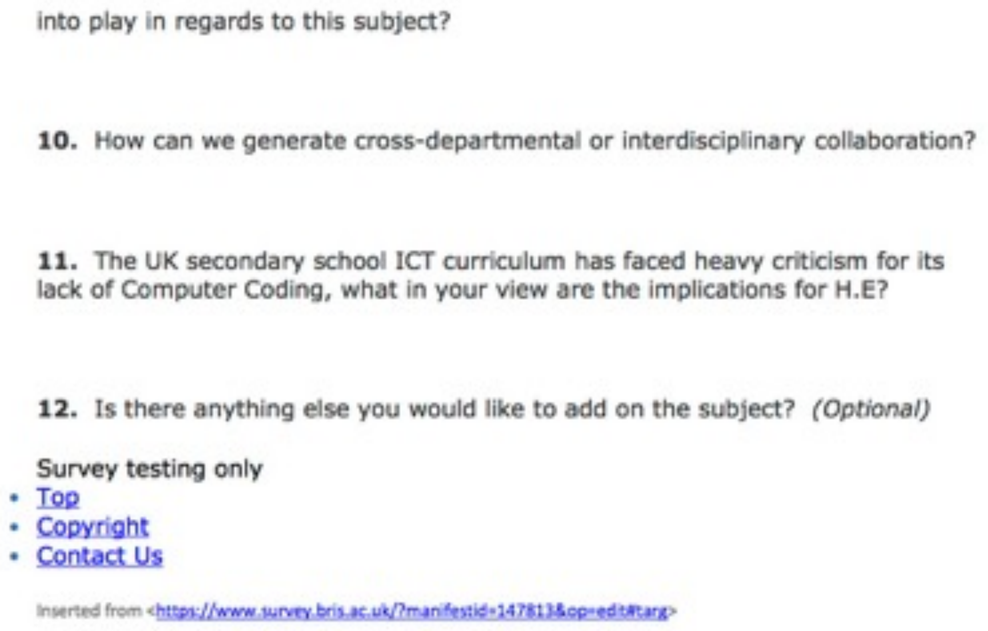
Really challenging – not just computing learning coding – lots of challenges. It only becomes possible when areas put aside their mantra and strip it down to what it is they are trying to do. Find and embrace the disciplines. Find projects and themes or subjects that can be embraced that are not dependant on their discipline. The more you can get to contemporary media themes and tie things together through social media (the way it is changing behaviour – society and culture)**HOW IS SOCIAL MEDIA CHANGING CULTURE and society – the more it is embedded the harder it is to identify it – the more pervasive it is**, through media through pervasive media. Through contemporary cultural existence , economics – if you can tie it together , if you can pull out the discussion round that – that applies to all areas – engagement with that story. Where they chose to take it is up to them. **Open THINKING Open minded .** Social media as a phenomenon**.** Its important to understand what new technology is in order to steer it into more interesting directions.

It doesn't have to involve technology – it doesn't have to involve screens and keyboards.

Relay information that is important to them ***The conversation here moves into more abstract concepts*** This gets people more excited. SCHMIT AT GOOGLE SAID THE LOVIES AND THE TECHIES AREN:T COMING TOGETHER INT HIS COUNTRY. ***FIND PEOPLE WITH INTERESTING IDEAS*** The way design is taught and the way art is taught are two different worlds. Design is technical .

Q UK Secondary

It has big leaps to go. When raspberry pi came out – excited by intention – you need to make it cool – secondary education pupils – what they don't know about social media isn't worth knowing. There are engagement in it in an intense and full on way. There is a coolness – it attaches itself to certain identities and sub cultures. Trying to get coding or hardware development into that is the trick. If the raspberry pi can get into the young people lives that is the most exciting way. IT departments are doing some good wiring up kind of things but it is not driven in the same way the art department might be driven but that is also quite mark making i.e. drawing etc **This is interesting – IF we don't; generate images and we don;'t program apps or websites what do we make?** As a community of research and creative practice better student could be welcomed



into play in regards to this subject?

10. How can we generate cross-departmental or interdisciplinary collaboration?

11. The UK secondary school ICT curriculum has faced heavy criticism for its lack of Computer Coding, what in your view are the implications for H.E?

12. Is there anything else you would like to add on the subject? *(Optional)*

Survey testing only

- Top
- Copyright
- Contact Us

Inserted from <https://www.survey.bris.ac.uk/?manifestid=147813&op=edit#targ>

but as a sector or curriculum we may not be ready to receive these students with better coding

Architecture
Architecture
Architecture
Architecture
Architecture
Architecture
Audio and Music Production
Ba fine art
BA Graphic Art
BA Hons Graphic Arts
Fashion
Fashion Communication
Fashion Communication
Fashion communication
Fine Art
fine art
graphic arts
Graphic Arts
Graphic Arts
Graphic Arts
Graphic Arts
Graphic Arts
Graphic Arts
Graphic Arts
graphic arts
Graphic Arts (Graphic Design)
Graphic Arts BA: Interactive
Graphic Design
Graphic Design and Illustration
Graphic Design and Illustration
Graphic Design and Illustration
M.Arch
Ma Fine Art
MArch
MARCH
MArch, Masters in Architecture

Masters of Architecture
MRes Art and Design
PhD Art and Design

2. Level			
4:		2.5%	1
5:		12.5%	5
6:		27.5%	11
7:		10.0%	4
8:		10.0%	4
Graduated:		37.5%	15

3. Coding Experience			
None:		47.5%	19
Beginner:		45.0%	18
Intermediate:		7.5%	3
Advanced:		0.0%	0

4. As an Artist or Designer what are your thoughts on the use of computer coding?
All for it. I think that art should reflect the modern computer led times. I think it is a great tool and should be integrated more in the arts. I think there is still room for some more expansion in the field.
Although the learning curve is steep and therefore it is underappreciated, coding definitely has a place with design.
As a designer, coding is becoming a major of what we do. I personally have taken an interest because I feel that it will benefit me later on in my career and make me more employable. There is a great emphasis at the moment on being a 'jack of a trades' within design.

As an Architect, computer coding and the use of Building Information Modelling is essential in the procurement and execution of designs. Whilst nothing, in my opinion, can replace the intimacy of putting pencil to paper, the use of CAD is useful in designs for several reasons including: Highest degree of accuracy. Particularly in relation to BIM, there's nowhere to hide. It can be so easy to forget or miss something, using the computer model as a tool means that the design can be correct first time around with nothing missed off. Speed of information gathering and production.
As an Illustrator, computer coding is something which i have personally never really ventured into. There are instances when i have used some forms of coding, for things such as isadora or arduino, but nothing major. Another instance where i would possibly look into it would be when building a website, and this is probably where it would be most useful for me. Knowledge about computer coding would be handy, but in the field i am in, it is not a requirement.
Before the workshops and stuffs, there was a student who used coding to do web designs and things. I always found that impressive and fun to look at. I like the idea of having codes to make something work, is something I can never think of like, is unbelievable of what coding can do. I can understand how coding would make a website work, but coding in the Arduino software to make an LED work? Is just amazing! I would have normally thought is all about wires work and electricity, but is all about coding!
Coding provides a very interesting results/advantages. Generative arts etc and also having something other than the standard set of tools of a graphic designer.
Computer coding allows another dimension to be added to a designers skill in creating an environment.
Computer coding itself is slowly becoming more accessible to students with programs such as processing bridging the gap between to the world of Java , but still a programming language barrier remains. Programs like Max MSP have tried to create a more user friendly concept for designers to create with, what is most important is that the most flexible and creative functions of raw code are made available to user and this seems to be a challenge for software designers.
Computer coding, to me, feels like a useful skill for a designer. You can get by without it with the use of sites such as Cargo or software like Muse but in terms of employability some knowledge or previous use of code is a plus. Coding seems to allow a lot more possibilities than things like cargo, and offers a lot more individuality to web design.
Could be very useful, however the building we are houses in in university is a prime example how computer systems dont work in a building so, i have been put off using intergrated systems within a building until perhaps they have been imporoved
Digital design packages such as Photoshop or Quark are a fantastic means to an end. However any given software package is just that - a package containing a finite number of tools. The ability to code means that these tools can be added to, altered or even ignored altogether. Being able to code is like being able to make your own bespoke paper, paint and brushes.
Having started HTML/CSS coding with the simple use of PNG images in my 2nd year of UNI for a portfolio project I changed my option in order to pursue Interactive work in my Third Year. Had it not been for coding I would now be a penniless illustrator.
I am a bit of both so i guess As a Designer ; I use computer code every other day and i think its an amazing unrated tool that more designers need to use. As a Artist ; Code allows me to go above and beyond what i want to create and is always bringing me ahead of my competition.
I believe art ought to represent the time in which it is conceived. Computer coding must be considered a fundamental influence on the designer and, even if unused, ought to be acknowledged in the studio. Too often projects exhibit a superficial understanding of coding and in doing so undermine the value of ideas.
I believe the perception it is extremely complicated done only by those with ICT degree or the like to either create computer programmes, and more recently apps. I never thought it could be used in the architecture field, other than by those coders back at autodesk etc who write programmes for architects.
I don't know what computer coding is

I dont know much about it to be honest, however it is something that i would love to learn or have the chance to learn. I have seen a few online videos about how it can potentially be used, and it is very inspiring, but it also seems a bit out of reach.
I felt it was a very interesting aspect which can allow you to design and create in different ways. I would encourage anybody within the design field to explore computer coding and see the different ways in which they can incorporate it in their work.
I find it interesting yet hard to learn it by myself. It is really rewarding when you finish something yet you might not be sure it will work at the start. That might be the reason why some people get discouraged from computer coding.
I have been told that knowledge of basic coding is essential to anybody and although I'm not aware if computer coding has any direct impact on designing, I will be willing to give it a try.
I have little knowledge and understanding of computer coding, however from the little understanding I do have on the subject, I suppose computer coding could possibly be used in an interesting manner in order to produce a set of commands that in turn produce something in an art form as a result of a repetitive or maybe a completely random commands.
I have never needed to use coding myself (preferring analogue methods) and would say I view it from afar with a mixture of curiosity and vague foreboding. As we move forward and coding proliferates into more areas of the workplace (Art and Design or otherwise) I think there is almost an increased pressure on designers to be a Jack of All Trades - I don't think there is going to be an excuse not to learn it in future. I don't necessarily think that is a bad thing. The possibilities coding can open - especially if we take the viewpoint that code itself is a form of craft - are almost endless, which is something that should definitely be explored by anyone with an interest in it.
I have never used computer coding before but would like to learn as I think it could be very useful.
I have no idea about it, really don't have a clue what its all about or how to do it.
I think coding can be an extremely useful and rare skill which a designer can have. I think it can be useful to get work and hired at an agency due to the fact that a lot of studios may need coding work doing for them.
I think coding is important in order to maintain an up to date web presence.
I think for the time we are in now, where most things a very computer interactive, computer coding is quite important and is need to adapt a chosen art form with the current time. For example, having a fashion catwalk with added elements of computer coding to create a more visual show.
I think it is important as a designer due to setting up your own website in your own design to display your work, create an online portfolio or become a freelance designer.
I think you can do some really amazing things with it.
It seems complicated at first but there are a variety of standard layouts that allow an easier understanding of computer coding.
it seems to becoming more and more important for designers to be able to do a bit of coding - or at the very least understand a few basic principles so you can work with a web developer and design for web with the constraints of web in mind. it seems like a daunting thing to approach but i've heard code academy is a good place to start.
Much of what we do as designers involves the consumer/end user being engaged with what we are trying to do. Any advance in technology which can help us achieve this should be utilised fully in my opinion - I am a big fan of collaboration between fields. The interactive elements within computer coding can also help gauge public opinion about the project.
My experience on computer coding is very limited. I have had one module on my course on coding. I found it quite confusing in many aspects, i was given a template for all of the information i had to insert which made it easier but i found i didn't learn much from it.
None whatsoever.

Personally I don't know anything about coding. I would though love to understand and know more about it as I feel my practice could be enhanced by knowledge of computer coding.
Personally, as a student of architecture, I think that caad softwares are creatively limited and ideas have to be brought, from physically drawing and model making, to a computer for adding finishing touches or possibly pushing complex geometries further. Maybe coding would allow architecture students to be creative with computers? Current softwares take control of ideas where maybe an understanding of coding would give the student control.
The language is very alienating and it feels a world away from what an artist might do, although that is subjective. Perhaps it just doesn't feel like you can be expressive with coding, but it can be used as a tool to creative something that might be expressive.
Very beneficial
very useful tool, anyone who knows how to seems to be asked constantly for advice

5. What experience have you had with computer coding?
A little, working on web portfolios
Absolutely none, but I would be willing to learn as I feel the digital world only opens more windows of opportunity.
Arduino, isadora and website building.
Coding changes your whole perspective on design and helps you to focus on the practical elements behind the scenes. This then lead me to explore CSS3, HTML5, jQuery, ASP.NET and I am currently learning the basics of C# whilst building online print forms and interactive databases. I am currently Junior Developer at a Design Firm due to be promoted to Developer within the next 3 months.
creating website portfolio and basic use in flash
For the module i had to create a server for a network in which a client could access media content. Another part of the assignment i created a website with hyperlinks to access the media content on a network. For this i had to use coding to create the website.
I am always curious about new stuffs so when the workshop was available I was enticed to give it a try and see if I could do some experiment with it and perhaps use them in my projects. I have had experience with coding from Arduino, Dreamweaver, Using the inspector section in cargocollective to change the layout of my website. I find it easier to play with coding in the inspector section in cargocollective, because they are easier to use, and our tutor Chris were teaching us simple stuffs that could be creatively used in laying out the images on a website, and making gifs. When I had the workshop for Arduino, it was a whole new experience to me, and I have to say I enjoyed it but mainly because there was instructions and tutorial provided by Andy. If it was just a kit who someone bought for me, I would never touch it because is not something I am dare to do it myself just in case I make a mistake. It was an interesting process how a software with entered codes through a cable makes something work physically. We made LED lights shine in different ways, make a fan rotate, I can't really remember but there were several small devices we used to connect to Arduino. I had workshops with dreamweaver too but that was ages ago, and after the 1st workshop I never got back into it, I realised is not within my interest really.
I built my website (with help) and used Isadora and Arduino a little bit, but not a lot.
I have coded Arduinos, Flash and the CSS for my Cargo Collective Page. I have also recently started to learn jQuery and HTML on Code Academy.

I have some experience from before studying Fine Art Ma and Graphic Design as I was studying physics where it was important to understand C++ for the purposes of solving complex equations. I haven't finished the course, yet I was familiar with the basics of computer languages. During my Ba in Graphic Design there wasn't any course in programming except for some Joomla and WordPress modifications.
I have used computer coding to create websites.
I once knew someone who worked for a digital marketing company whos job it was to write code.
I studied Computing at AS-level in college, but my class was unfortunate enough to be told that we "would not have to write a program in the exam", and so not much attention was given to actual coding. (1/3 of the exam was a 20-mark question asking us to write a script for something or other by the way -.-) After that I didn't see any coding again until my degree, when i started using actionscript in Adobe Flash, and I saw some html in Dreamweaver too. Some time studying Arduino also brought in more hand-on coding and electronics.
I used coding a little when altering a cargo theme by 'inspecting the element' and tweaking things in there to get the results I wanted.
I was attending arduino lessons. This was the first time I had ever been involved in computer coding and could easily make the link between my current projects and the applications of this programme.
i went to a couple of sessions at university but it was a quick skim over website design in general, so I havent learned alot of coding specifically yet
Mainly used CAD software for the last three years at Uni despite having lessons and CAD sessions at uni. The main driver for me using CAD was getting a placement during my year out in an Architect's office which accelerated my CAD skills massively. As a direct result, my computer skills for my postgraduate degree were so much better than before, and made my work so much better.
My first experience with code was in year 9 of high school when i was 14, copying website source codes and manipulating them in textEdit on a windows 98 (the good old days). My main experience with computer code is building web sites using HTML/CSS/PHP/JAVA this is for small clients that run clothing boutiques (www.atqm.co.uk) to larger sites for modelling agency's (www.nemesisagency.co.uk/) and artist ran blogs (www.holdthenovelty.com). My other main use of code is as an artist where i use code for building applications for screen visuals for festivals like Parklife and smaller events such as OWT Discourse.My main code as an artist is JAVA using Node based apps like Isadora,Apple Quartz Composer and Modul8.
Non
none
None
None
None
None
none
None what so ever in recent years, however a few years back I taught myself HTML coding to create a blog.
None!! My projects so far have been involved with the basics of our interaction with space. Technology, for me, is another layer on top of this which can be used to heighten our experience of a place, especially creating points of interest within it. I am interested to use it in the future as a method to display my work more effectively.
None, i've only had beginner experience with web coding and have never had an interest to pursue computer coding, there would have to be a very good reason for me to go down that route.
None.

None.
None.
None. I have given lectures on the crossovers between craft and technology, and how we can view one as the other.
Not a lot. I do handle codes and scripts but these are rudimentary so I mot often calculate these using paper mathematics. I do however transcribe my mathematical designs into software to perform modelling and repetitive functions.
not much at all. I am sure that in school (back in 1990's I learnt how to manoeuvre a small robot around the floor using directions, however it all then was lost after that and i haven't had to use it much at all. (shame really).
Nothing untill university
Some HTML, flash action scripts and basic PHP but i am just getting started with Java at the moment
Throughout my last year at university i have attempted to code three separate websites with a lot of assistance from tutors. Two of the sites were coded from scratch and i feel i have learned a fair amount of basic coding from this.
Very basic coding; html, css, arduino and processing.
Very basic use of Java Script.
Very little but I havent needed to use it for past projects.

6. Has anything stopped you from learning or advancing with computer coding?
Alienating language and i have found no good reason why i should learn it.
Apart form the intense code sessions when learning code during the bitter hours between 1am - 5am nothing has crossed my mind to quit coding.I still use the same tools i did when i was discovering code at 14 so theres no barriers stopping me all you need to code is textEdit and reference books. I dont aspire to become a web design/developer so in time i will code less and less as a career.
busy university schedule, not having the contacts
Firstly, its scary, kind of like learning a new language so that puts a barrier up straight away. I've worked a lot on changing the cargo layout and played a little making a website from scratch but with a lot of help.
Frustration and it becoming Too dry, trying to learn the code side without a practical application.
Heavy schedule with my Architecture course.
I found out I was dyslexic in my Third Year but somehow this had enabled me to move further with code as I identify better with the mix of letters and symbols. I do have to triple check my spelling just in case though on every job.
I have had a few attempts but feel like i have learned more each time. There are a few useful websites which attempts to break it down and make it easy but it can be quite overwhelming.
I have no idea what could possibly be generated to benefit an architectural idea through coding.
i think a large part of it is that i advance my learning better in a class setting rather than on my own- motivational issues!

I think one of the main things missing at the moment is a program that incorporates everything into a single GUI in a similar way to programs such as dreamweaver with html, but obviously on a more expansive multi-programming language basis, learning within this sort of environment is always quicker and the provided examples and syntax always make creating easier.
I think the resources for learning how to code are quite accessible, so nothing has ever stopped me.
I thought it was only for those with ICT degree. I have friends who have a degree in computing (not sure what degree exactly) and I have seen some of their coding and it seems extremely complicated to a layman. I do not no what other use a computer code has other than to write software programmes.
In the Architecture industry, there seems to be 'families' of software, Graphisoft which uses ArchiCAD and Artlantis to render with, Autodesk which has AutoCAD and 3Ds Max aswell as others such as Rhino and Cinema 4D. I would say that, by advancing my skills whilst at work and uni in one family of software has put me on the back foot in terms of getting a job with practices who use a different one. I wouldnt say this has stopped me from progressing my skills, I think its more that I need to progress my skills but wont have the time to do this whilst I have other commitments.
In university, time. Outside university, not knowing where to get started or where to begin
It is difficult to explain the subject to people without prior theory or knowledge of the subject. This is the situation i was in which hindered my learning. with all the technical words and phrases that are in the subject of computer coding it is essentially like learning a whole new language.
It isnt really something which is required in illustration, and i have always found coding quite intimidating.
It seems to be something which I would find very difficult to learn, therefore have never attempted.
Laziness really. I like it, and think its cool, but its not my main area of interest in design. I would like to do a bit more of it though, understand it a bit better.
no, I just have never thought about it. it is not something that has ever been talked about or offered.
No, I really want to learn it but time hasn't yet permitted me to do so.
No, not really, it was just hard to start and then on it was just about my own patience and interest to learn it. There are tutorials and videos that can be found on the internet to help you learn, nevertheless I believe that they are usually useless if you have no background in coding. I believe that the way to learn best is still through structured course.
No. The facilities in the art and design academy have been a good platform for me to engage with computers. I would say that the principals of all software education is that one must have the responsibility to learn for oneself.
Not being aware of it.
Not knowing how or where to learn about it.
Not really, I have no desire to pursue a course on the subject.
Not understanding it. I am not great with computers, maths and all that.
Nothing has stopped me from learning exsept myself and thinking that its something I may not nessaserally need to learn.
Nothing in particular, it's more a lack of immediate returns. Some kind of online starter training or book for beginners would probably be enough for somebody to pull themselves into coding as a hobby, but there is definitely a lot more which could be done with kids in school to get them comfortable with the basics. As for advancing, some kind of club or project-driven social media group would be a great way to encourage budding coders to stick at it.
Only the lack of will to learn it. I have focused my practice on traditional print methods, and haven't really needed to use code.

The amount of time required to get a grounding / steep learning curve. The need for using maths, which most creatives would be glad they have far since distanced themselves from.
The busy fast paced nature of the architecture degree course leaves little time to pursue extra curricular interests. Personally I would have loved to have played with the DVJ's but the constant deadlines didn't allow the time to fit things like that in.
the difficulty of not knowing where to start, which language to learn. I only tried to learn when i had to use it for my web portfolio.
The lack of staff available to develop learning.
The one thing which I found had stopped me from advancing with computer coding was time. I had so much going on that I felt that there was just not enough hours in the day to learn more. Maybe if this was included somewhere within a module it would allow more time to be spent on it.
The only thing I would say is we are not exposed to it enough before higher education. If we had a brief knowledge of what is possible and how to achieve it then I may have been more likely to use it. We have so much going on in the degree learning how to draw, design, integrate systems etc. that it may seem like one more thing to learn about. The facilities at LJMU are incredible and as of yet I don't think we have utilised them fully, which is an error on our part! I certainly will be during the summer time, trying to increase my knowledge of what is possible!!
The only thing stopping me up until now was time, however i would love to learn how to do it now i have some spare time from university.
We have never had the opportunity in university to learn about computer coding, although we are encouraged to create an online portfolio.
Yes, I think is mostly my disinterest after the workshops and the fear of not being able to succeed on anything if i were to able to use them in projects. I am scared of coding, there are so many rules and different things and because i don't have a good memory, so I never step further in learning computer coding.
Yes, not knowing anything about it.

7. How much about computer coding, if any, would you like to learn?
A little to make my own website.
Anything that would help me with fashion communication and gaining skills that I would need in the future.
As much as possible
as much as possible really - so i could confidently make a fairly basic website from scratch through to completion
as much as possible, its getting more and more important to learn for graphic designers
Basics
basics, enough to allow me to make architectural models in the future, eg, lighting systems in time with the music or something
Basics.
Depends on what i was aiming to achieve, if for a project i had to pursue it then i would spend the time to learn the basics. It would probably feel like learning to use Photoshop or Flash for the first time, i had good reason to learn these programs, less so with computer coding.
Enough to build my own website and be able to edit it and understand the 'language' of code.
Everything

Firstly on the development of my own personal online portfolio/website and then how it could enhance my personal set of skills for further job potential.
Having mastered CSS and HTML capabilities I am now moving into jQuery and C# but as language travels well over different types of coding this tends to be easier once you practice the basics. There are still days even now when I find something completely new to work with so I can imagine this to continue for years to come.
I already know some very basic coding and feel i would greatly benefit from learning more.
I answered this a little in the last question. For me, there is probably loads that I didn't even know is possible and I will be trying to gain experience during the summertime.
I intend to use the summer to further my skills in order to advance my design capabilities and to create a website to showcase my works.
I like to mnow some more stuff on web coding, so I can keep updating my website and play around with it more a bit more independently.
I think for my future it would benefit me greatly by learning a lot more about computer coding. As much as i need to know for my industry/ subject would be fine.
I wish to learn html and css which I am already familiar with and continue with learning Java script and Java itself, maybe some of C++
I would like at least a working knowledge of html/css and the coding for arduin/processing.
I would like to have a basic understanding
I would like to learn as much as possible, but mainly in the areas of jQuery, HTML and CSS.
I would like to learn computer coding for designing my own website and blogs
I would like to learn how to set up my own website and design the pages.
I would like to learn more about computer coding for webdesign.
I would like to learn some basics that could be applied to projects.
I would love to be able to build my own website from scratch without use of templates and so on.
I would love to learn alot! I would love to know how to make websites and understand how computers work, as i personally think it is the future. I think they should teach it in schools now-a-days.
I would love to learn in the future more about how MySql databases communicate with PHP code and how they work with eCommerce shops and large blogs.I would also want to master JAVA and get to grips with music visualization and projection mapping using code as part of being a artist.
I'd like to know what it is and if I'd benefit from using it.
If I could, I would like to learn as much! I would like to learn about the coding for websites, coding for arduino and devices to make my own creative things!
If it can benefit my design, personal development skills or employability, then yes I would be interested. If it didn't, then I wouldn't be interested.
if there was a work shop I would probably go, I guess its a useful skill to have.
MORE WEBSITE STUFF
Now that I have learnt the basic skills needed to use this programme I will continue to progress these after graduation. I would like to learn as much as possible about computer coding as I think that it is relevant to any field that I will work in.
Same as above, This would depend on me gaining an understanding of what the potential outcomes are for an architect through an understanding of coding.

Shit loads!!! I am always looking to learn more about the programs I use so that I can use them to their full potential. Recently I have been looking at GDL script which is writing computer scripting for generating 3D objects in ArchiCAD which is both confusing and interesting.
The basics would be nice to start with.
The first thing that springs to mind is 'enough to make games', i love making things that people can interact or play with. I suppose by that standard though the next question would be "how complex do you want your game to be?". I think seeing something you've made which looks 'professional' gives a tremendous sense of accomplishment, so the first step is probably making something which performs a task, seeing it work, and giving it a smart-enough front end.
Would be interested in learning the capabilities and the possible outcomes of computer coding and to begin with the basics of how to go about putting something together to achieve and end result.

8. What benefits if any do you see arising from gaining knowledge of computer coding?
A base knowledge in any field can always prove useful.
A new medium to express your ideas and possibly even to gain ideas from.
Advanced job opportunities as I would have more experience in a different area. But as previously stated I do not know how this would be relative to an architecture degree.
Alot for future careers
Any skill is useful to have, computer coding is probably near the top of useful skills if you were working in an environment with computers, someone might be more likely to employ you.
As fully graduated architects we have at most 5 minutes to grab a clients attention. By utilising a more interesting and engaging method of presentation, with light and sound interaction, then there is more of a chance that we will interest more clients and get more work!
Being more clued up on the matter and being able to rely on others as much in computer and technical matters.
Creating work which can be taken to the next level using computer coding and creating something which someone without the knowledge of computer coding wouldn't be able to do.
Currently I am the only student from my course to become a developer. Having been offered Jobs abroad almost immediately after Graduating I imagine a long and fulfilled career. In comparison some of my peers have yet to gain a single commission or job in their field.
Gaining skill that most other people from whatever course may not have possibly giving you the upper hand.
gives another option to creating personal projects, looks great on the cv even if you know a bit.
Greater understanding of technology
Having control over how you release your work into the world (i.e. not relying on tumblr or cargo collective type sites) would probably be the most rudimentary benefit to learning to code. Although with a real interest in the subject I think the use of coding to generate artwork is very interesting. E.g. Brendan Dawes' work.
I can only imagine that coding would give a more technically driven architect opportunities to create more sustainable architecture through design and maybe directly through technical functions of buildings?
I feel that when making models, computer coding can help to make yours stand out and become more interactive to the user.

I think a lot of good can come from learning computer coding. People would be able to create things which suited their own work individually without resorting to shaping their work to fit an existing software or structure. Also, i think we would see a lot more personalised websites, not just the same type of templates used over and over.
I think if you know how to code, it will give you more flexibility in designing when facing projects. It gives people another medium of producing artwork. And if someone knew how to do webdesigns, it will allow more creativity in the design of the layout, so that there would be much more than just a variety of styles. Computer coding is eventually essential to the always advancing technology in today's society, who knows if someone master computer coding and invent a new kind of code system connected to many devices it will benefit society in many ways, especially in welfare and medical care. I see coding as a language too, one of the benefits is that everyone around the world will understand it and it would be very flexible for people to work together of different ethnics, bring more ideas and concepts together to create devices that helps humanity and the brain.
I think more job opportunities would definitely open up.
I think such knowledge will broaden my potential in design and presentation and so increase my chances of success in my career.
I think there are two main areas where I will benefit. Firstly being able to achieve exactly what I want from the pieces of software, making my life and job easier. And secondly, practices at the moment are looking for potential employees who know all about the software [particularly BIM which is taking off massively in the profession] who can improve the CAD capacity of their practice. Also the RIBA Plan of Works for Architects, the work stages by which a building progresses, has recently been overhauled to integrate BIM into it, which is an example of how computer coding and programs are driving the profession and making it evolve to embrace these new technologies.
I think there might be better opportunities to work with new technologies in the industry i am in, and also gain a better understanding of how the computer programmes that i use nearly every day work and function. I would also like to be able to make my own websites and online resources.
I would be able to be more creative when designing things like a web page for my work.
I'd probably be able to solve problems better to do with computer programmes etc.
If i ever had my own business computer coding would help me create a business website. The more knowledge i have from computer coding the more professional my website would/could look.
It is beneficial to learn coding; for a Graphic Designer to be able to create and manage web pages and other interactive visual content. For an artist it might be useful for developing art pieces that include interactivity or creating art pieces that have the ability to serve as "useful" art. Also a lot of art has shifted towards using on-line galleries and interfaces to view art wherever on the world simultaneously.
It would make me more employable, especially with the ever growing online/digital world, if I can understand and design using it.
it would open more doors once graduated.
Learning a new medium to play with that will let you do things you can't in another format.
learning something new
Learning to code can give a whole new dimension to my work, as it gives me another platform to work with, aside from print, etc. Coding can also improve my future prospects, like I said above, there is an emphasis on learning these things in the design community.
Lots, you can do work for yourself and maybe work for others. Its never going to be a dying business.
More eligable for visualisation jobs.
More job opportunities
Presentation skills, not having to outsource, another skill to add during interviews

The ability to make stuff that does stuff :D It means freedom, and in terms of creative software, a truly blank slate to work with. No, a blank nothing into which you can put whatever kind of metamorphic rock you like.
The benefits are certain, A designer who can apply code to projects can create more varied work than the designer who cannot.
The benefits of standing out with amazing new work in the creative industry and also the main benefit of making your life easier.I has to witness students tile images hundreds of times in massive photoshop files over graduation , if only they knew that they could do all that work with a few lines of Java in processing and it will be 100% perfect.
The university would benefit from students setting up online portfolios, I think it would contribute to the success of graduates.
Understanding computer coding is becoming more and more important. As a designer I understand that my work will probably take digital form more so than any physical printed work. Technology provides us new ways to communicate almost every day with new apps and new devices bringing new formats within which to design. Websites, for example, continue to develop new ways with which to communicate with their visitors through the symbiosis of code and aesthetics. A wealth of opportunities lie in being able to understand both the design of aesthetics and the code that builds a foundation behind it.
understanding how to design for web; more employment opportunites ie in design jobs where they say some basic coding skills are desirable - it gives you the edge over your competitors

9. What problems do you think art and design students could experience when learning computer coding?
Alienating language and they would need a reason to learn it or really enjoy it for the challenge.
Being too lazy and deciding its too hard.
computer programs required
Confidence. "I can't do maths" "It makes no sense to me" "It's too complicated" "It's dead nerdy innit?" are all preconceptions, people need to be introduced to it almost without being told what it is that they're doing. Literally anybody can code if they're shown how, and people need to understand that before they write themselves off.
Design students tend to have a higher rate of disabilities such as Dyslexia, This in turn could help or hinder them. As a creative mind I imagine students similar to myself would struggle with more of the practical elements of coding in comparison to the creation of something visual without coding.
From my own experience, I think that student's might find it hard to remember all the codes and stuffs, of course it is not expected for them to REMEMBER everything, however practise makes perfect, and computer coding might be a simple thing when learning at the beginning, however as one step higher it becomes super complex already. Other than that. I can't think of anything else really.
Having a more creative mind rather than a computer/technology mind could cause some problems but I think they both can be adapted together to create a final project/result.
having to learn in a new way
I don't know.
I think a lot of arts students get very quickly put off at the idea of learning code because they think its all numbers and math and you need A* level math to do it.Also with the arts students you get a lot with dyslexia like my self as its a visual subject and the idea of making an image with punctuation and numbers is scary.

I think personally computer coding is intimidating to art students, myself included, because as artists we tend to like freedom to be creative and expressive. Although you can be creative with coding, i think coding in itself looks rigid, and has to follow certain structures and rules, which to many visual people may seem quite boring.
I think some people will find it daunting, and sone will find it hard. I also don't think some people see the relevance of learning this and it's use in their particular field.
I think students are reluctant to learn coding because they believe it to be difficult. The teaching of coding in schools often sensationalises the challenges when actually such hurdles are better tackled by the student. To learn coding requires a disciplined and singular focus and to develop an idiosyncratic understanding of systematic methods.
I understand there is a debate whether computer coding is an art form or scientific method. For me it sways on the scientific side of the fence. There is therefore more chance of more 'artistic' minded people being put off the idea of using a slightly mathematical method in designing something - even if the outcome is perceived artistic. Luckily I am interested in science and believe that a basic knowledge in it can greatly improve our creative minds. We can't ignore science in our endeavour to 'create' things, experiences and buildings.
I'm not sure really. Guess it's quite daunting at first but there's no reason for it to be problematic.
If I were to compare it to the teaching of any caad packages, it just needs to be about building from the absolute basics.
It can seem extremely complicated learning all the terminology and language to begin with.
It could be to much of a regimented process, some creative and artistic individuals may not enjoy facing boundaries.
It is seen as too complicated and some don;t see the practical application.
It requires a large input at the start for very little gain, most students would be put off my the amount of learning required to even start to write basic code.
It seems a little intimidating at first, which can put people off. Also I think it takes a while to be able to understand it well enough to be able to fix problems with out help or start more exciting experiments.
Lack of concentration / will, I expect.
lacking time i suppose, and in lacking time it is difficult to find the courses. I would love to have the courses built into the architecture curriculum, but in a way that makes it more fun. We have ICT classes for sketchup and things, but often they are not as inspiring as they may like them to be and i found them very difficult to learn from. If classes were to be put on i would suggest it was a summer thing and they ran for a week during the summer, and then when we started back in the september we would be able to crack on and show off what we have learnt.
Maybe seen as computer coding according to my understanding is based on calculations and algorithms, art and design students may find it restrictive and possibly hard to grasp. I personally come from a physics and mathematics a-levels background and find the subject intriguing however I find architecture and design a challenge. Somebody from a complete art and design background will possibly be the opposite and find something like computer coding a challenge.
N/A dont know enough about art and design.
Not having good enough experience with computers and probably not having enough patience.
Not learning enough as there simply isn't enough time unless it is taught during level 4. It could also cause problems if students say they are competent at computer coding when they apply for a job, but really they only know the basics.
Not sure
Not sure
Not sure. Maybe the level of complication it involves.

Not their field of expertise. Coding strikes me as a complicated 'art' in itself, of which I would not be sufficiently educated to enter at even a medium difficulty level, or have the sufficient basic skills required to even understand coding in its most primitive form.
One of the hardest things i found with arduino was the actual coding sequences. By this I mean the wording for the sequences.
Technical understanding of how coding corresponds to a function.
That it is very tedious and not stimulating
the technicality of it can get a bit mind boggling at times - too much in a short space of time can be overwhelming and its easy to get lost if you dont keep up. detailed and clear handouts are a must.
the time it takes to create and edit code, finding little mistakes in a load of words and numbers is very frustrating at times, especially when you have deadlines to hit.
Traditionally people who have a brain geared towards creativity may have issues learning programming languages. I think this will always be a problem and until either a self explanatory language is created or possibly a GUI within which the code can be interpreted by the designer as its result in real time rather than in its coded form.
Unsure
Well for artists it might be difficult if they don't have any knowledge of computer coding, or have the lack of knowledge of what the technology does or doesn't offer them.
When I first arrived at uni, I wasn't computer literate at all, and the prospect of CAD was very daunting at first. The main driver for me to learn and adopt it was to make my work better and to give myself half a chance of getting a job at the end of uni. I think the main issue it to do with confidence and the almost fear of what things do within a software package. This is quite difficult to overcome, and in my experience the classroom environment didnt help me much. I learned alot from one on one sessions at work and most of all, self-teaching myself at home using books, youtube tutorials and the like. I think that alot of this 'fear' will be overcome in the next few years as computers etc are so commonplace, most first years are now completely at ease with programs such as Photoshop, which I didnt really touch until my third year.

7.6 Gatekeeper Information Sheet

LIVERPOOL JOHN MOORES UNIVERSITY
GATEKEEPER INFORMATION SHEET

Title of Project

Digital Integration in Art and Design Education: Engaging Students with Computer Coding.

Name of Researcher and School/Faculty

Researcher: Andy Freeney
Liverpool School of Art and Design / Arts Professional and Social Studies

You are being invited to allow the staff and students of your school to take part in this research, but it is important that you understand why the research is being done and what it involves before you decide to take part. Please take time to read the following information. There are contact details at the bottom should you wish to ask any questions or to clarify anything that is unclear to you. Take time to decide if you want to take part or not and feel free to ask for more information.

1. What is the purpose of the study?

The research project you are engaging in considers making computer coding a more fundamental tool for artists and designers in Higher Education. Computer coding is reaching a point in its evolution where it is entering the public domain and could arguably open up many new and exciting possibilities for Art and Design students even at the beginners' level of competency.

Through working with students over the past few years I have met those who, in their final year, want to create projects incorporating computer coding but find that they lack the knowledge or time to do this successfully. Not surprisingly they see 'Computer Programming' as another language with a steep learning curve. This is especially true if the student has zero or limited previous experience. Without the proper support one set back can result in the abandonment of ideas that have the potential to lead to exciting and inspiring projects.

I hope to correct this gap in knowledge by generating ways to help students to engage and succeed earlier during their time at university.

2. Do staff and students have to take part?

No, no one has to take part. This research is completely voluntary. Should staff choose to take part you will be contacted to arrange an interview. Should they not wish to attend an interview but still take part an alternative online questionnaire will be available.

Should students choose to take part they will be emailed a link to an online questionnaire which is completely anonymous.

Everyone is still free to withdraw at any time. No one will be recorded as having withdrawn, or as having taken part as no personal date of any kind will be kept.

3. What will happen to staff if they take part?

- They will be asked to attend an interview. This will be recorded and transcribed.
- Before conducting the interview they will be asked to sign a consent form.
- Should they not wish to attend and interview you can fill out an online questionnaire instead where they will be asked for their name and organisation.
- The questionnaire will take approximately 30mins to complete in depth but could be completed in less time.
- They do not have to answer every question.

4. What will happen to students if I take part?

- They will be emailed a link to a questionnaire with 10 questions on it.
- Should they choose to complete it, they give implied consent for your answers to be used in the research.
- The questionnaire will take approximately 30mins to complete in depth but could be completed in less time.
- They do not have to answer every question.

5. Are there any risks / benefits involved?

There are no risks or benefits of this research other than you will have contributed to the research which aims to gain a better understanding of the field.

6. Will staff and students taking part in the study be kept confidential?

Yes, any of the opinions used in the research will be anonymised. No personal data will be kept.

Contact Details of Researcher and Director of Studies

Name: Andy Freeney
Address: Liverpool School of Art and Design, Duckinfield St, Liverpool, L3 5RD
Email: a.freeney@ljmu.ac.uk
Phone: 01519041106

Name: Peter Appleton - Director of Studies
Address: Liverpool School of Art and Design, Duckinfield St, Liverpool, L3 5RD
Email: P.Appleton@ljmu.ac.uk
Phone: 01519041175

7.7 Glossary of Terms

Adobe - Adobe Cor Adobe Systems. developed the Creative Suite (CS) enabling graphic design, video editing, and web development to be done on a computer.

Arduino - Arduino refers to an open-source electronics platform or board and the software used to program it. Arduino is designed to make electronics more accessible to artists, designers, hobbyists and anyone interested in creating interactive objects or environments.

ArchiCAD - ArchiCAD is an architectural software application offering computer aided solutions for handling all common aspects of aesthetics and engineering during the whole design process of the built environment.

ASP.NET (Language) - ASP.NET is an open-source server-side web application framework designed for web development to produce dynamic web pages.

Audacity - Audacity is an open source stereo enabled sound editor.

Autodesk - Autodesk builds software that helps people imagine, design and create a better world.

BIM - BIM is an acronym for Building Information Modelling, or Building Information Model. It describes the process of designing a building collaboratively using one coherent system of computer models rather than as separate sets of drawings.

C (Language) - C is a high-level and general purpose programming language that is ideal for developing firmware or portable applications.

C++ (Language) - C++ is a high-level programming language and adds object-oriented features to its predecessor, C.

C# (Language) - A hybrid of C and C++, it is a Microsoft programming language developed to compete with Sun's Java language.

Cargo - Cargo collective in an online content management system that enables the presentation of visual content while incorporating good standards of design.

Coding (The process of programming a computer using a programming language or computer language)

Computer Coding (See coding)

Computer Language - A programming language designed to communicate instructions to a machine, particularly a computer.

Computer Programming - A process that leads from an identification of a computing problem to executable computer programs build uses a programming language.

Computer Vision - Computer vision is a field that includes methods for acquiring, processing, analysing, and understanding images and, in general, high-dimensional data from the real world in order to produce numerical or symbolic information,

Creative coding - Creative coding is a type of computer programming in which the goal is to create something expressive instead of something functional.

CSS (Language)- CSS stands for Cascading Style Sheets. CSS describes how HTML elements are to be displayed on screen, paper, or in other media.

Director - Adobe Director (formerly Macromedia Director) is a multimedia application authoring platform created by Macromedia and now managed by Adobe Systems.

Electronics - A branch of physics and technology concerned with the design of circuits and manipulation of electricity.

Flash - Adobe Flash (formerly called Macromedia Flash and Shockwave Flash) is a multimedia software platform for production of animations, browser games, rich Internet applications, desktop applications, mobile applications and mobile games.

GDL (Language) - In computer-aided design, Geometric Description Language (GDL) is the programming language of ArchiCAD library parts.

HTML (Language)- Hypertext Markup Language, a standardised system for tagging text files to achieve font, colour, graphic, and hyperlink effects on World Wide Web pages.

Isadora - Isadora is a proprietary graphic programming environment for Mac OS X and Microsoft Windows, with emphasis on real-time manipulation of digital video.

Java (Language) - Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible.

JavaScript (Language) - Javascript is an object-oriented computer programming language commonly used to create interactive effects within web browsers.

Joomla - Joomla is a free and open source content management system (CMS) designed to assist users in building websites and other online applications.

jQuery (Language) - Query is a concise and fast JavaScript library that can be used to simplify event handling, HTML document traversing, Ajax interactions and animation for speedy website development.

Ludics - Ludics focuses on the concepts of play and games.

Max/MSP - MaxMSP is a visual programming language that helps you build complex, interactive programs without any prior experience writing code.

MUSE - Adobe Muse is a product by Adobe Systems. The software is focused on allowing designers to create websites without having to write any code.

Modul8 - Modul8 is software designed for performance and real time video

MOOC - MOOC stands for “Massive Online Open Course”. It is a course of study made available over the Internet without charge to a very large number of people.

New Media - A means of mass communication using digital technologies such as the Internet or digital video.

openFrameworks (Language) - openFrameworks is written in C++ and is an open source toolkit designed for "creative coding".

PHP (Language) - PHP is a script language and interpreter that is freely available and used primarily on Linux Web servers. PHP, originally derived from Personal Home Page Tools, now stands for PHP: Hypertext Preprocessor.

Processing (Language) - Processing is an open source programming language and integrated development environment (IDE) built for the electronic arts, new media art, and visual design communities

Python (Language)- Python is a widely used high-level, general-purpose, interpreted, dynamic programming language.

Quartz Composer - Quartz Composer is a node-based visual programming language provided as part of the Xcode development environment in Mac OS X for processing and rendering graphical data.

Tinkering - An experimental form of open source hacking or electronics development. An attempt to repair or improve something in a casual or desultory way.

Snippets - A snippet is a programming term for a small region of re-usable source code, machine code, or text.

SQL - SQL stands for Structured Query Language. SQL is used to communicate with a database.

Wordpress - Wordpress is a free publishing software and online content management system (CMS) that is open-source allowing developers to create a wide array of plug-ins, themes and widgets.