UpStage Team

Investigation of Programming Languages

UpStage 2015 S2

Table of Contents

[Introduction 2](#_Toc432849962)

[Programming Language Requirements 5](#_Toc432849966)

[C# 6](#_Toc432849967)

[Java 9](#_Toc432849968)

[Node.js 12](#_Toc432849970)

[Python 15](#_Toc432849971)

[Proof of Concept 17](#_Toc432849972)

[Rationale for choosing C# 20](#_Toc432849976)

[References 22](#_Toc432849977)

[Appendix 24](#_Toc432849978)

[Steps for prototyping in Java 24](#_Toc432849979)

[Steps for prototyping in C# 24](#_Toc432849980)

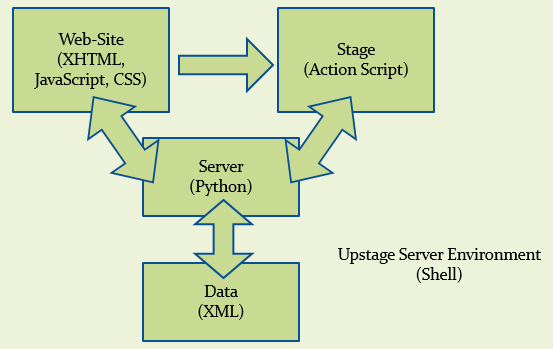
|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Changes |
| 1 | September 21, 2015 | Alyssa Byun, Jing Han, Joshua Kartono, Siatua Uili, and William Stokes |  |
| 2 | September 30, 2015 | Alyssa Byun, Jing Han, Joshua Kartono, and Siatua Uili | * Introduction added * References added * Research evidence on Atmosphere and SingalR added * Issues addressed from v1 feedback fixed |
| 3 | October 16, 2015 | Alyssa Byun, Jing Han, Joshua Kartono, and Siatua Uili | * Research requirements edited * Introduction edited * Document reformatted * Appendix added * Issues addressed from v2 feedback fixed * Changed rationale (from C# to Java) |

# Introduction

During the spike before the project team starts developing the new UpStage, we should decide which programming language to use for developing it. The existing UpStage has been developed with Python. The clients have said that the team does not need to stick with Python for the new UpStage. They also told us that they will be satisfied as long as the team completes this project and delivers a high quality product.

For the frontend of UpStage will be written in HTML5, CSS and JavaScript similar to the existing UpStage. This document will focus on the backend language of UpStage.

## Current Version of UpStage (Farrell, 2013).



The existing system for backend was developed in Python that runs in the server side. Action Script was also used for flash and the frontend of the system was written in XHTML, CSS and JavaScript. For the data storage, the data is stored in XML files.

One of UpStage requirements is to be open source for the public wishing to contribute and the program needs to be modular.

At the moment it uses Flash’s Action Script which supports UpStage’s many functionalities (i.e. viewing multimedia including streaming videos and audio (Adobe Flash Player, 2015)). Unfortunately Flash is becoming obsolete as it is largely being replaced by HTML5 and it does not support mobile platforms which is needed in the new UpStage. (Minnick & Ttittel, 2014).

On the server side we use Python’s Twisted and if we are to develop a new UpStage we will not be able to reuse the python code even after changing the Action Script side so it would be best to find a language for the backend.

## Programming Language Requirements

For the research of programming languages, we have identified several criteria so that we are able to measure the difference between the chosen languages for researching.

For better development of UpStage

* There must be good testing and debugging tools for the programming language so that we can reduce the defects.
* The programming language must be learnt easily by the developers because UpStage is open source and also an ongoing project so it will be passed down to many more students in the future.
* The desired functions of UpStage must be well-implemented in the programming language. There are a few functions that must be considered when choosing programming language for UpStage. It should support
  + Real time web functionality (asynchronous communication) for the stage including the chat function and the performance
  + Text-to-speech
  + Video streaming
  + Concurrent drawing
* The programming language must be available across the platform that UpStage uses.
  + The program must be run on Linux server.
* Is it open source

## Candidates

The chosen languages to research are listed below with the reasons for choosing.

* **Python**

The current UpStage is written in Python. Some code can possibly be reused and the team should learn this language to maintain the current UpStage. We would like to see how Python would fare compared to the other languages.

* **Java**

All team members are familiar with Java as BCIS students in AUT are taught programming papers in Java, and also the new members joining every semester in the future are.

“Given Java’s rock-solid foundation, IDE support, threading capabilities, debugging features, and vast array of libraries, it is still regarded by many as king in the world of programming” (Snyder, 2015).

* **C#**

As familiar programming languages like Java can save some time for learning and getting used to, we considered other familiar programming languages. One of the team members is an avid user of C# through her work experience and has suggested that C# and ASP.NET framework is good to build a web-based system.

* **Node.js**

One of the past team members, Yue, has researched Node.js and suggests it as a good alternative to Python’s Twisted. Yue has produced a prototype using Node and has found that using this language for the new UpStage supports a lot of functions required such as the drawing tool. Since it is written in JavaScript it is fairly easy to learn.

Node.js has “a strong following with an impressive amount of documentation, tools, and best practices considering its age, and is rapidly gaining adoption among enterprise IT organizations” (Snyder, 2015).

All members might want to use the programming language they are most familiar with, but what we need to consider is whether it is the best programming language that is suitable to support the required functions for this product with an acceptable quality.

# Programming Language Requirements

For the programming languages:

1. Testability  
   - Are there any useful testing and debugging tools for this language?
2. Learnability  
   - How much knowledge does the team have of this language?  
   - Is it easy to learn?  
   - How much prior knowledge does it require to use this language?
3. Performance  
   - Can it run in real time (asynchronous communication)?  
   - Does it support the chosen architecture?  
   - How fast?
4. Availability across the different platforms.

* Must support UpStage platform (i.e. Linux)
* Is open source throughout its use on UpStage?

# C#

C# is defined as a simple, modern, general-purpose, type-safe, object-oriented and high-level programming language developed by Microsoft running on top of the .Net Framework (Benton, Cardelli, & Fournet, 2004).

**Readability**

Advantages

* C# uses the curly brackets like Java does. It is more readable than other languages such as Python that do not use the curly brackets.

Disadvantages

* LINQ in C# makes easier to code but harder to read.

**Testability**

Advantages

* .NET provides many testing tools for many purpose. I.e. Telerik Test Studio for a UI testing, TypeMock for unit testing, etc.
* ASP.NET has a useful tools like Elmah (Error Logging Modules and Handlers) or Glimpse (the open source diagnostics platform of the web to inspect web requests to reduce debugging time).

**Learnability**

Advantages

* Like Java, C# provides a number of features to make it easier for a developer to code in this language such as type checking, bounds checking, uninitialized variable checking and garbage collection.
* Language is simplified for some functions such as multithreading.
* Easy to learn if you are familiar with other programming language like Java or C++.
* As UpStage requires real time web functionality, SignalR can be used. SignalR is a library for ASP.NET developers that makes it incredibly simple to add real-time functionality to the application (“SignalR”, n.d.).

**Performance**

Advantages

* Have all the benefits of .NET framework as C# is tied to .NET framework.

**Availability across the different platforms**

Advantage

* It was not supported on platforms other than Windows, but .NET Core became open source in 2014. Roslyn, ASP.NET 5, Entity Framework, etc. these are now all free to use (“.NET Blog”, 2014).

Disadvantage

* However, if the platform is not Windows, the knowledge of using .NET framework on other platform such as Linux is required as C# is a language more native to Windows. It also does not have a lot of support on

**SignalR**

SignalR is an open source library for ASP.NET developers that simplifies the process of adding real-time web functionality to applications. SignalR provides a simple API for creating server-to-client remote procedure calls (RPC) that call JavaScript functions in client browsers from server-side .NET code. SignalR also includes API for connection management (for instance, connect and disconnect events), and grouping connections.



It handles connection management automatically, and lets users broadcast messages to all connected clients simultaneously. It also supports server-push functionality, in which server code can call out to client code in the browser using RPC, rather than the request-response model.

# Java

Java is a popular and widely supported programming language. It follows the WORA (Write Once, Run Anywhere) philosophy, which means the Java Virtual Machine can interpret any java code, ensuring that as long as the JVM is present, the code will be able to be run. For this project, in terms of options for programming the Back-End of UpStage, Java has the following advantages and disadvantages.

**Testability**

Advantages

* Forced/checked exceptions mean that they will not be left as an afterthought, reducing maintenance costs in the future and making it easier to debug the program. This is important as new team members come and some team members leave every semester, which means less people with knowledge of the current code and more people who need to understand the code. Checked exceptions will help reduce the amount of bugs, which new team members will not know about if these are not done.
* Backwards compatibility is an integral part of Java, which will reduce maintenance costs in the future as functions will be less likely to break or stop working between versions or updates. The JVM, or Java Virtual Machine, is fully backwards compatible and can run any code compiled with previous versions of the JDK. This means that any recent browsers can be used to open UpStage without having to program specifically for them.
* Java has access to excellent and powerful testing tools such as Junit. This will aid programmers in debugging, or otherwise testing, the code. Automated testing tools such as Selenium is also compatible with Java. This is useful for UpStage as being able to automate testing and test creation will save time, and the UpStage project is time intensive.

Disadvantages

* Although programmers are forced to catch exceptions, they are not forced to handle the exceptions, and may instead mask them by an empty exception handling, which will make the code harder to debug or maintain. This can make future members confused because an exception is thrown but the source or cause is not stated. This can be avoided by forcing team members to handle all caught exceptions.

**Learnability**

Advantages

* An extensive library of modules and APIs lessens the chance of having to “Reinvent the Wheel”.
* Familiarity, as Java is the main language taught to AUT students, and as such upskilling time for new members can be reduced.

Disadvantages

* A vast library means it can be difficult to find the proper libraries and/or tools, which costs time. An inferior option may be used in place of a better option if not enough research is done in choosing which libraries to use, as some libraries are known to be of worse quality than their equivalents.

**Performance**

Advantages

* There are several libraries, such as Atmosphere and Spring Framework, which aids java in achieving asynchronous communication, which is a requirement for the project.
* As Java depends on its JVM to run the code, its performance is relatively stable on all platforms (“Learn About Java Technology”, n.d.), whereas C# might run slower on non-windows platforms, and Python and Node.js are dynamically typed, which will slow them down on run-time (Ishizaki et al., 2012). This is especially true when the amount of data is not large.

Disadvantages

* Java applications may have some performance spikes, such as when a class is loaded when it is first referenced, or when the garbage collection happens, which might result in non-asynchronous communication for a short period of time. As such, java’s real-time performance is not consistent. This will not come into play when the amount of data is not overly huge, as is the case with UpStage.

**Availability across the Different Platforms**

Advantages

* Widely supported on most platforms, including computers and mobile devices, like Python and Node.js, but unlike C# which is primarily made to work on windows platforms.

**Atmosphere**

## Atmosphere is a real-time framework for Java which allows asynchronous communication between two or more instances of a web application that uses the client-server architecture. It uses WebSockets to achieve asynchronous communication but also has fallbacks in cross-browser operations.

It supports most popular browsers, such as Firefox, Google Chrome, Internet Explorer and Opera, and also supports both iOS and Android, which is a requirement for UpStage as it has to be able to run on all popular platforms. If Java is used to develop the backend for UpStage, Atmosphere can be integrated with Java code by simply adding the necessary Atmosphere jar files as the project’s dependencies in the IDE.

The main advantage of Atmosphere for the UpStage project is that it works with all popular browsers, and the real-time functionality is ensured on all platforms without having to explicitly program for some platforms. This means the project team only has to come up with one solution, saving time and resources.

# Node.js

Node.js was introduced in 2009 for Linux so is open source and is a server-side and browser technology written in JavaScript. It is a platform independent runtime environment for creating web applications and it uses the event driven architecture making it suitable for scalable real-time applications (Node.js, 2015).

**Testability**

Disadvantage

* JavaScript has a lack of features to assess modules or the code quality

Advantage

* Four types of tools needed for successful Node.js testing:  
  A Testing Framework (Mocha, Vows, Intern)  
  An Assertion Library (Chai, Assert)  
  Stubs (Sinon)  
  Module Control (Mockery, Rewire) (<https://github.com/mochajs/mocha>)

**Learnability**

Advantages

* NodeJS apps are written in JavaScript so it is easy to learn if you are familiar with JavaScript. It is a neutral language if there is an uncertainty in choosing Java and C# as most web developers should be familiar with JavaScript
* Tutorials available for new users of node showing easy to follow steps in creating http server useful to UpStage in starting up a new version for students new to node (Cannaday, 2013). Also there are social networks for Q and A apart from video tutorials dedicated to Node. (<http://nodetuts.com/> - for nodeJS video tutorial)
* Desktop IDEs for Node.js that are free to use and familiar to AUT students are: Eclipse IDE and NetBeans.

Disadvantage

* UpStage development support, for upskilling or for other reasons, in node may not yet be available as it is still developing and is not as mature as languages such as C# and Java.

**Extendibility**

Advantages

* For server development and web development
* Because it is written in java script it is easy to make the transition from modifying front-end to back-end code. For two totally different languages used (for back-end and front-end) it would be easy to make silly coding mistakes, for example missing the braces in JavaScript and semicolons in python (Kaplan-Moss, 2011).

Disadvantage

* Node.js structure is not mature enough to identify any packages that can be trusted

**Performance**

Advantages

* Uses V8 developed by Google for use in Chrome and it compiles and executes at a high speed due to its use of single-threading for asynchronous programming (Sychay, 2014)
* Non-blocking I/O
* Asynchronous programming: Ability to handle thousands of concurrent connections with minimal overhead on a single process compared to creating multiple threads that may show different results for different users of UpStage. This would be useful for the UpStage server.
* The best architecture candidate chosen for UpStage at this point is Event Driven Architecture and Node.js uses an event-driven model making it a great language for real-time applications (“Node.js”, n.d.).   
  In terms of architecture patterns, node also has frameworks for MVC (“Node.js”, n.d.).

Disadvantages

* Doesn't support multi thread.
* The main advantage of node.js is the speed/performance aspect (asynchronous programming). This is not very useful due to the small user base in UpStage (making speed the least important factor in a language needed) and because the main issues the current version has (using Python’s Twisted) are testing and maintaining the code. There are some free tools identified in this research.

**Availability across the Different Platforms**

Advantages

* Node.js is designed to work on many operating systems such as, Linux, which is used for the current UpStage server. It is also runnable in OS X, Microsoft Windows, FreeBSD, NonStop, IBM and a few more which give a lot of options to UpStage. But Linux will still be use for the new UpStage and it is open source (“Node.js”, n.d.).
* It is open source. Suitable for UpStage development as we need it to be open source for other UpStage contributors

# Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Python is simple and easy. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

**Advantages**

* Python is a high level dynamically typed language. It is easy to pick up for new programmer and easy to use. If people are used to Python, it actually more efficient in developing stage. For developing same functionality, by using Python or other languages, the code written in Python is shorter.
* Example for simple print out sentence:

In Python:

print "hello world";

In Java:

public class Main {

public static void main(String[] args) {

System.out.println("hello world");

}

}

* For IDE, Netbeans 8.0.2 started supporting Python which is a bit easier for AUT students (We all familiar with Netbeans)
* For current UpStage, the code is written in Python and we need to maintain the current UpStage. We must learn Python. If we use Python for our new UpStage, we don't need waste extra time for learn other new programming languages.
* Debugging Python programs with PyDev is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on.
* Our clients know a bit more about scripting languages (like Python) than other programming languages. The code will be more readable for our clients if we use Python. Our clients could getting a bit more involved during the development phase.

**Disadvantages**

* Python does not have brackets for functions, if the code is hundreds or thousands lines, python will a bit a hard to read without proper comments.
* Python is an interpreted language. It runs slower.
* Python 3 is not backwards compatible. (The current UpStage is using Python 2.7.3) If we are going to use Python, we need to make sure the consistency of the version of Python.
* Absence from mobile computing and browsers. "Python is present on many server and desktop platforms, but it is weak in mobile computing; very few smart phone applications are developed with Python," says Carbonnelle. "It is also rarely seen on the client side of a Web application." The most important part for new Upstage is that it should supporting mobile devices. For this reason, Python is not a good candidate for the Upstage, because we need to support mobile device for the Upstage.

**Testability**

Advantages

* Python has its own tools for unit testing called: unittest. It is the test module in the Python standard library. Its API will be familiar to anyone who has used any of the JUnit/nUnit/CppUnit series of tools.
* The Doctest is another testing tools for Python. The doctest module searches for pieces of text that look like interactive Python sessions in docstrings, and then executes those sessions to verify that they work exactly as shown.
* Doctests have a different use case than proper unit tests: they are usually less detailed and don’t catch special cases or obscure regression bugs. They are useful as an expressive documentation of the main use cases of a module and its components. However, doctests should run automatically each time the full test suite runs.

**Learnability**

Advantages

* Python is a high level language and dynamically typed language, it is easy to learn and easy to use.

**Availability across the different platforms**

Advantages

* Python can used in Linux/Windows/IOS Mac X. Python can be used in most popular platforms.

**Performance**

Disadvantages

* Python is a scripting language, it runs slower than other structure languages. (e.g. Java, C#)

# Proof of Concept

The project team has investigated on four programming languages and whether those candidates are suitable to be chosen as a programming language to build UpStage. Java and C# have been selected as appropriate languages for UpStage development based on the research criteria to measure and judge the programming languages. However, it was not easy to choose the best only one from those two languages, because both seem incomparably suited for UpStage. Therefore the team decided to make a very simple version of UpStage. As a prototype is a demonstration of what is actually feasible with existing technology and where the technical weak spots exist, the developers are able to understand better about the development environment (Bernstein, n.d.).

As UpStage is a platform for cyberformance, it is important to consider how to implement the real time web technology. Some of team members prefer to choose C# and some prefer Java. The individual’s preference of programming language is subjective but still important. The team thought that is important for everyone to experience both languages to try to be fair. Therefore, we work in pairs and make two parts for the prototyping so that after completing the first part, the pairs can exchange the project to complete the second part. For the first part of the processes, we record the steps of implementing the function so that we can compare the simplicity of the two languages (See the appendix section). Then we can also see the other differences between those two programming languages.

## Process

* Part 1: Simulation of aspects of chat function

Two pair programmers work with the task to create a simple chat app in  
different programming language.

* + C# – Alyssa and Sia
  + Java – Jing and Joshua
* Part 2: Simulation of aspects of showing avatar

The pairs exchange the programming language to extend by adding an image  
button that adds or removes an image in real-time.

* + C# – Jing and Joshua
  + Java – Alyssa and Sia

## Criteria

1. Ease of Use  
   - How easy to code?
2. Readability  
   - How easy to understand the code written by other members? (whether they follow the coding standards)
3. Extendibility  
   - How easy to extend the code?
4. Responsiveness  
   - Always responsive?
5. Real-time  
   - How well it works when there are more than two clients at the same time?

## Findings

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Prototype Criteria: | | | | |
| Language | **Ease of Use** | **Readability** | **Extendibility** | **Responsiveness** | **Real-time** |
| C# | There wasn't much use of C# in the prototype but SignalR (the API for chat) was fairly easy to use. The pair programmers involved with chat were familiar with SignalR so it was easily implemented. The second pair completed the extension in a short period mainly because JavaScript was the main language involved so this only shows that C# is not any better for coding only that C# easily supports implementation of this prototype’s functions. | The steps to complete the prototypes were documented and made it easier to understand what had been done. However when extending the code it was difficult to understand coding. In C# the unfamiliar layout made it hard to read. | Adding the avatar button was easily done | Was always responsive | yes |
| Java | Most of part 1 consisted of researching for a suitable API for the chat function (Atmosphere was used). Chat was easily developed but after atmosphere was used it became difficult to use the code. This is because Atmosphere added a large amount of code within the projects classes. Using its commands were simple but the structure of the classes changed a great deal confusing the students about where to make changes to add the extra feature (adding avatar button). This process took the team hours and a lot of effort and in the end failed to produce the functionality required. | In Java it was the Atmosphere's code that was difficult to read. | The code was difficult to extend due to the complex coding in the Atmosphere API (used for chat). | Was always responsive | The chat prototype was real-time but extension (i.e. adding avatar button) was not real-time. In the end it was not solved. |

# Rationale for choosing Java

Before the project team starts developing the new UpStage, we need to decide which programming language to use for developing it. The existing UpStage has been developed with Python. The clients have said that the team does not need to stick with Python for the new UpStage. They will be satisfied as long as the team completes this project and delivers a high quality product. All team members are familiar with Java as BCIS students in AUT are taught programming in Java, one of them is used to C# through her work experience and some of them knows a little bit of Python as they have been working on the maintenance of the existing UpStage. All members might want to use the programming language they are most familiar with, but what we need to consider is whether it is the best programming language that is suitable to generate the required product with good quality.

Each team member has chosen one programming language and researched on it to find out its advantages and disadvantages based on several criteria we have decided, so that we can decide which language is suitable for the new UpStage. The researched languages are Java, C#, Python and Node.js. Each of these has its advantages and disadvantages, but we want to discuss why we should use Java and how it will work for UpStage.

Java is an open source programming language which follows the Write Once, Run Anywhere (WORA) philosophy. It is platform independent, and code written in Java can run on any platform once compiled. This is a big advantage as UpStage needs to be able to run on common platforms such as Windows and Linux, and mobile devices. As UpStage is a web-based application used for online performances, the performers’ performance must be seen by the audience immediately without a page refresh. A technology that can make real-time web applications is required for UpStage and Node.js is a technology commonly used for developing real-time web applications. Of course, other programming languages also have tools that provide real-time function such as Spring for Java, twisted for Python or SignalR for C#.

Ease of testing is a big part of the reason we chose Java. Java has access to powerful testing tools such as Junit, while C# has access to an excellent IDE with powerful debugging tools in Visual Studio. Python has plug-ins for existing IDEs such as PyDev for the eclipse IDE. They allow testing of Python code. Successfully testing Node.js code requires getting and learning how to use 4 components, namely the testing framework, an assertion library, stubs and a module control tool. This is unnecessarily complicated, which is something we want to avoid.

Learnability is another factor to consider. Node.js is written in JavaScript, which is a relatively simple language to learn. Some of the papers taught at AUT also use JavaScript, which means it is likely that current and future team members have some knowledge on JavaScript already, and thus reduce the amount of upskilling needed. The front-end of the current UpStage is also written in JavaScript, which further increases its advantages. Team members should already be familiar with Java as it is the main language taught in AUT, so it also requires minimal upskilling. Python is a required learning regardless of the language we decide to use, as the current UpStage uses Python, and it needs to be maintained. However, for future team members, when the current UpStage no longer needs to be maintained, Python will no longer have this advantage. C# requires the most upskilling compared to the other 3 languages, as it is not taught in AUT.

In terms of performance, Node.js is ideal as it is relatively lightweight, unlike Java and C#. It is also faster than those languages, because it does not utilize blocking threads and thus do not have too much downtime. Also, unlike Python which is interpreted during runtime, it does all processing beforehand, thus ensuring that performance is consistent during runtime. While Java can be faster in certain cases, it is prone to inconsistent performance as classes are only loaded when first referenced, or garbage collection can happen at random times, which can cause performance spikes. Node.js also has non-blocking I/O which ensures smooth performance during runtime. However, this will only come into play when an extremely large amount of data needs to be process, which will not happen with UpStage.

Java is also open-source and available on all popular platforms, including Linux which is a requirement of UpStage. Unlike C#, which was designed to primarily work on the Windows platform, Java is compatible with most browsers and mobile platforms by default, meaning there is less work that needs to be done in making it compatible with various platforms.

There are other advantages we did not mention here but it is worth to choose with these reasons. As developers ideally want to create small and fast micro-services that are easy to maintain, Java is a good choice for its familiarity and testability. It will help the team make a great application that is a good quality and easy to maintain. Node.js can still potentially be used for certain parts of UpStage if needed, particularly parts which require consistently high performance or do not require rigorous testing, but Java is the main language that will be used.

# References

Adobe Flash Player. (2015, October 13). Retrieved October 15, 2015, from Wikipedia: https://en.wikipedia.org/wiki/Adobe\_Flash\_Player

Atmosphere (n.d.). In *Wikipedia.* Retrieved from <https://github.com/Atmosphere/atmosphere/wiki>

Benton, N., Cardelli, L., & Fournet, C. (2004). Modern concurrency abstractions for C#. *ACM Transactions on Programming Languages and Systems, 26*, 769-804.

Bernstein, L. (n.d.). *Importance of software prototyping.* Retrieved September 29, 2015, from <http://condor.depaul.edu/sjost/hci430/documents/prototypes-general/BernsteinPrototyping.htm>

Cannaday, B. (2013). *An absolute beginner’s guide to Node.js*. Retrieved from [http://blog.modulus.io/absolute-beginners-guide-to-nodejs](http://l.facebook.com/l.php?u=http%3A%2F%2Fblog.modulus.io%2Fabsolute-beginners-guide-to-nodejs&h=GAQHzDwihAQGCXWLyhiJbzi5MvY42SYGOrX-AyGXSiRB7Fg&enc=AZNHloibMH99FrLnHKKioURtOqUsnsUTvOysn8br-Q3xWpQ_jICcfNTxswwlyKk1MW39lj7byAB9fG49OaCNMBj1YcnCi3Tljop5wTfX3YtjAM6zySGooi2jLEZXIcZM6CzK8XXOBu-KRRpyYNK2JAK8tgxPXhI9a01_yeyVT1NxdQ&s=1)

Farrell, C. (2013). *Induction Presentation* [PowerPoint slides]. Retrieved from AUT UpStage SVN Repository <https://dctsvn.aut.ac.nz/ustage/Documentation/2013%20-%20Semester%201/Induction%20Plan/>

Ishizaki, K., Ogasawara, T., Castanos, J., Nagpurkar, P., Edelsohn, D., Nakatani, T. (2012). *Adding Dynamically-Typed Language Support to a Statically-Typed Language Compiler: Performance Evaluation, Analysis, and Tradeoffs* [PowerPoint Slides]. Retrieved from: <http://www.cl.cam.ac.uk/research/srg/netos/vee_2012/slides/vee18-ishizaki-presentation.pdf>

Java (programming language). (n.d.). In *Wikipedia.* Retrieved from  
<https://en.wkipedia.org/wiki/Java_(programming_language)>

Kaplan-Moss, J. (2011, January 7). RE: What are the benefits of developing in Node.js versus Python? [Web log comment]. Retrieved from <https://www.quora.com/What-are-the-benefits-of-developing-in-Node-js-versus-Python>

Kennethreitz (2014). Testing Your Code. Retrieved from: <http://docs.python-guide.org/en/latest/writing/tests/>

Kopp, M. (2011). *The impact of Garbage Collection on Java performance*. Retrieved from  
[http://apmblog.dynatrace.com/.../the-impact-of-garbage.../](http://apmblog.dynatrace.com/2011/03/24/the-impact-of-garbage-collection-on-java-performance/)

Krill, P. (2015). *A developer's guide to the pros and cons of Python*. Retrieved from <http://www.infoworld.com/article/2887974/application-development/a-developer-s-guide-to-the-pro-s-and-con-s-of-python.html>

*Learn About Java Technology* (n.d.). Retrieved from: <https://www.java.com/en/about/>

Minnick, C. & Ttittel, E. (2014, April 30). How Adobe is moving on from Flash to embrace HTML5 [Web log post]. Retrieved from <http://www.cio.com/article/2376661/internet/how-adobe-is-moving-on-from-flash-to-embrace-html5.html>).

.NET Blog. (2014). *.NET Core is open source.* Retrieved August 19, 2015, from <http://blogs.msdn.com/b/dotnet/archive/2014/11/12/net-core-is-open-source.aspx>

Node.js. (n.d.). Retrieved from <https://nodejs.org/en/>

Node.js. (2015, October 15). Retrieved October 15, 2015, from Wikipedia: <https://en.wikipedia.org/wiki/Node.js>

Npm (software)). (n.d.). Retrieved from <https://en.m.wikipedia.org/wiki/Npm_(software))>

Python (programming language) (n.d.) in *Wikipedia*. Retrieved From <https://en.wikipedia.org/wiki/Python_%28programming_language%29>

Reitz, K. (2014). *Testing Your Code.* Retrieved from <http://docs.python-guide.org/en/latest/writing/tests/>

SignalR. (n.d.). *What is ASP.NET SignalR*. Retrieved August 19, 2015, from <http://signalr.net/>

Snyder, R. (2015, August 25). Why Node.js beats Java and .Net for web, mobile and IoT apps [Web log post]. Retrieved from <http://www.infoworld.com/article/2975233/javascript/why-node-js-beats-java-net-for-web-mobile-iot-apps.html>)

*What is the difference between C, C++ and C#?* (n.d.). Retrieved from <https://www.quora.com/What-is-the-difference-between-C-C++-and-C>

# Appendix

## Steps for prototyping in Java

1. Use NetBeans IDE
2. If not using NetBeans, or Tomcat is not installed, install Tomcat 8.0.26 (Apache Server)
   1. Set JAVA\_HOME environment variable in catalina.bat by adding “set JAVA\_HOME = \*path to jdk\*”
   2. In the Command Prompt, navigate to the Tomcat /lib folder and run the startup.bat batch file and go to localhost to test that the server is working
3. Create new Java Web Application
   1. Select Spring MVC as the Framework
   2. Select Tomcat as the server
4. Install Atmosphere Libraries
   1. Download the Atmosphere runtime libraries
   2. Under the project, Right click the libraries folder and choose add jar
   3. Choose all the downloaded libraries and add them
5. Coding
   1. In the ‘web’ folder, create a folder for the Model (i.e. Javascript files). In this case, we made a folder called “Resources”
   2. The View is located in the ‘jsp’ folder, which is located in the ‘WEB-INF’ folder
   3. The Controller is located in the ‘src’ folder

## Steps for prototyping in C#

1. Use Visual Studio 2013
2. Open New Project – C# - ASP.NET MVC – name project and save
3. Install Signal R library - in Manage NuGet Packages
4. Make a Startup class - New Item – Owin startup class
5. Fill the configuration method in as required by readme file
6. Add script tag to the layout after rendering other script files (should be before SignalR)
7. Bundle signalR.js to be used in bundleconfig.cs
8. Add a Controller and View section. View section contains two input boxes and display box for the testing.
9. Include SignalR in reference.js to show alternative methods to use
10. In view – layout - $.connection.hub.start(); // to connect SignalR for every page
11. Index.cshtml will have javascript section
12. Create new hub class for chat – ChatHub – extends hub. This includes a sendMessage method which retrieves users input message from index.cshtml and calls displaymsg method
13. Index script section we have the methods to get user input and sends to server by calling (connection.chathub.server.sendmessage()) method. Create displaymsg method to send to client the nickname and message (displaymsg)