

Comparing today's most used web development languages

Arjan Assink

Due to the commoditization of internet connections, web applications are becoming more common as well. There are various ways of developing these applications, each with their own design methods, server properties and code conventions.

This paper is aimed at making a comparison between the three most commonly used web development options: PHP, Java and the .Net framework. These languages will be compared on four different qualities: deployability, performance, collective ownership and code re-use.

1 Table of Contents

| | | |
|----------|---|-----------|
| 1 | Table of Contents | 2 |
| 2 | Introduction | 3 |
| 2.1 | Attributes that will be evaluated | 3 |
| 2.2 | Project size | 3 |
| 2.3 | Method of testing | 4 |
| 2.3.1 | Experiment 1: programming assignment..... | 4 |
| 2.3.2 | Experiment 2: measuring performance | 5 |
| 3 | The languages | 7 |
| 3.1 | PHP 5 | 7 |
| 3.2 | Java Servlets..... | 7 |
| 3.3 | .Net Framework 3.5 | 7 |
| 4 | Comparison..... | 9 |
| 4.1 | Deployability | 9 |
| 4.2 | Performance..... | 10 |
| 4.2.1 | Experiment..... | 10 |
| 4.3 | Collective ownership | 13 |
| 4.4 | Code re-use | 13 |
| 4.5 | Grading..... | 14 |
| 5 | In practice | 16 |
| 5.1 | Hyves..... | 16 |
| 5.2 | TomTom | 16 |
| 5.3 | Findings | 17 |
| 6 | Conclusion | 18 |
| 6.1 | The best language | 18 |
| 7 | References | 19 |
| 8 | Appendix..... | 20 |
| 8.1 | Code | 20 |
| 8.1.1 | Javascript class..... | 20 |
| 8.1.2 | PHP class..... | 22 |
| 8.1.3 | Java class | 24 |
| 8.1.4 | C# Class | 27 |

2 Introduction

Due to the commoditization of the internet, web applications are becoming more common as well. There are various ways of developing these applications, each with their own design methods, server properties and code conventions.

This paper is aimed at trying to find out the best way of developing web applications today. This will be done by making a comparison between the three web development options that have become the most used over the last years, being: PHP, Java and the .Net framework. These languages will be compared on various qualities.

2.1 Attributes that will be evaluated

Each of the three languages will be judged on four attributes:

- Deployability - How easy is it to make the finished application available to the world
- Performance – How capable is each language in handling large amounts of traffic
- Collective ownership – How easy is it to work on one project with a large team
- Code re-use – How easy is it to use previously written pieces of code in a new project

These attributes are very important to modern day web applications. Because they are becoming increasingly more complex, demanding larger teams to work on them, the need for collective ownership and code re-use increases. Performance and deployability are of course important for any application that will be used.

2.2 Project size

The size of a web application might influence which of the three languages are the best option. In this research we separate small projects and big projects.

| Small Project | Big Project |
|-----------------------|-------------------------|
| Single server | Several servers |
| 1-2 programmers | More than 2 programmers |
| Development time of a | Continuous development |

few weeks | over a larger period of time

These two project sizes obviously don't cover all projects being done online; they are designed to cover the extremes for testing purposes.

2.3 Method of testing

Before such a comparison can be made, the languages have to be accurately specified, the flavour and version of each language that is used will remain the same throughout this paper. How each language is used will be motivated, based on conventions and advantages for each specific language.

Two separate experiments are needed to test all four attributes. The first experiment will be a programming assignment, made by various people in each of the three languages. The second experiment is aimed at measuring the performance of the languages.

2.3.1 Experiment 1: programming assignment

A test group will be given a programming assignment that has to be completed in each of the three languages. After the assignment, a questionnaire will be held to determine how well the test group judged a language.

2.3.1.1 The test group

The test group consists of four people with different backgrounds, to minimize the effect of their preference to a specific language.

| | | |
|----------------|-------------------------|--|
| Test subject 1 | Academic background | Bachelor of Computer Science |
| | Professional background | PHP Programmer (1,5 years) |
| | Programming languages | - PHP (1 year) - Java (0,5 year) - .Net (0,5 year) |
| Test subject 2 | Academic background | Bachelor of Computer Science |
| | Professional background | PHP Programmers (1 year) |
| | Programming languages | - PHP (1 year) - Java (0,5 year) |
| Test subject 3 | Academic background | Bachelor of Computer Science |
| | Professional background | .Net Programmer (3 years) |
| | Programming languages | - .Net (3 years) - PHP (1 year) |

| | | |
|----------------|-------------------------|---|
| Test subject 4 | Academic background | Bachelor of Computer Science |
| | Professional background | .Net programmer (1 year) |
| | Programming languages | - PHP (3 years) - .Net (1 year) - Java (1 year) |

2.3.1.2 Assignment

The programming assignment given to the test subjects is aimed at using the elements most common to web applications, retrieving and editing content from a database. An HTML layout will be provided, so all focus will be on programming in the language being tested.

First everyone will be asked to create a basis, using code they have previously produced, testing the code re-use of a language. After this has been done a second person will make modifications to the code produced by the first test subject, testing the collective ownership. Both test subjects working on each assignment will be asked to publish their code, testing the deployability.

2.3.1.3 Questionnaire

Because the three attributed that are being tested in this experiment have to be judged by the experience of the test subjects, this will be done using a questionnaire.

Questions in the questionnaire for the first assignment on how much of the code was written from scratch, and how much was used from previous projects. The questionnaire after the second assignment focussed more on how difficult it was to make modifications in someone else's code, and how much easier it would have been if they had done the first part for them selves.

Both questionnaires also contain questions on how easy it was to publish written code, to determine how well the deployability is. Also, questions will be asked on how clear the assignments were, to make sure this didn't influence the experiment.

2.3.2 Experiment 2: measuring performance

Measuring the performance of each languages takes a much more quantitative approach than the first experiment. A processor and memory intensive script will be written in each of the languages, testing for string comparison, integer operations, storage speed and printing text, all of which are very important operations for any web application.

A real world situation with a large number of clients sending requests to the webserver will be simulated in a setup that minimizes the effects of difference in network-speed and server specifications. This will be achieved by creating virtual machines for the webserver, preventing requests from actually having to travel through several networks, which might influence the final result. Also, by using virtual machines we can insure the tests are all run on a server with the same specifications.

Performance will be measured in the time it takes for the webserver to respond to each request with a completely rendered page. An average value of the response time will be calculated over a large number of repetitions to achieve representative results for each language.

The performance tests will be run with different numbers of clients making the requests, to see how each language handles the added load.

3 The languages

For testing purposes this paper is restricted to a specific implementation of each of the three languages. The choice for each implementation will be made based on what is most commonly used, or what maximizes the positive aspects of a language.

3.1 PHP 5

The PHP implementation that has been used for this research is the current stable release, PHP 5. This PHP will be accessed through the Apache 2 web server, and it will communicate with MySQL 5.0.

The reason this configuration has been selected is that this is the most recent version of the combination most commonly known as LAMP. The LAMP software bundle is considered to be the standard for PHP servers (Williams & Lane, 2004).

3.2 Java Servlets

Tomcat 6.0, based on Java Runtime Environment 1.5.0 will be the basis on which the Java Servlets for the research paper will run, and it too will communicate with MySQL 5.0, to minimize the difference with PHP. The web-server used will be Apache Tomcat, since this is one of the most popular web-server for servlets (Hunter & Crawford, 2001).

Ofcourse there are various ways in which Java can be used to serve web pages to requesting clients, but the deciding factor in using servlets was the efficiency and ease of use compared to other Java flavors (such as for instance Java Enterprise Beans or Java Server Pages).

3.3 .Net Framework 3.5

Data from a SQL server 2005 can be accessed by the .Net 3.5 Framework at the request of an IIS 7.0 server. Again, these versions are the latest in stable releases currently being used (Microsoft, 2008).

The choice for SQL server 2005 was made because it is the current standard, where in the past this might have been SQL 2000. An advantage of SQL server over Access is that the database can have it's own dedicated server instead

of having to be hosted by the same machine. The reason for not choosing MySQL is that it is not natively supported by .Net, and relying on third party connectors would be a disadvantage for .Net.

The .Net Framework offers a large variety of languages, the two most commonly used being C# and VB script. This paper will use C#, because this is considered to be the most powerful language by the community.

4 Comparison

As previously discussed, the comparison between the three languages in the paper will be done based on four attributes:

- Deployability
- Performance
- Collective ownership
- Code re-use

Each one of these attributes will be evaluated separately, comparing the languages to one another. By doing so, each language will get a grade, scoring them relative to the other languages.

4.1 Deployability

Deployability in this paper means how easy it is to transfer your code to a server, to make it accessible to the world. Of course this should be as quick and easy as possible for smaller projects, and as stable and robust as possible for larger projects.

Before code can be published to a server, the server must be up and running, and web-space must be configured. Since all languages offer a pretty manageable install method, there aren't any big differences on this part.

Assuming the server has been configured and is all done, the way PHP and .Net code are published are very similar. Both normally use a simple FTP client to get the code onto the server, which is very easy to use, and quick to set up.

An added feature .Net has over PHP is the ability to deploy the project's code in a .dll file, a compiled version of the code. This is particularly helpful in larger projects, because it benefits performance, and it's easier to deploy a single .dll file than a large quantity of smaller files.

Java is often published using the Ant Build system. This takes a little more time to set up, because a build-file must be made. Once the Ant Build system is in place, it is much less error-prone than a simple FTP transfer, which makes it a very robust and reliable system to use, great qualities when working with large projects.

4.2 Performance

Performance is measured in the time it takes the interpreter to respond to the request of a web-server. This excludes the time it takes the client request to reach the web-server, or the time it takes for the response from the web-server to reach the client, because these are tied in heavily with network speeds, which are not influenced by the performance of the web-language.

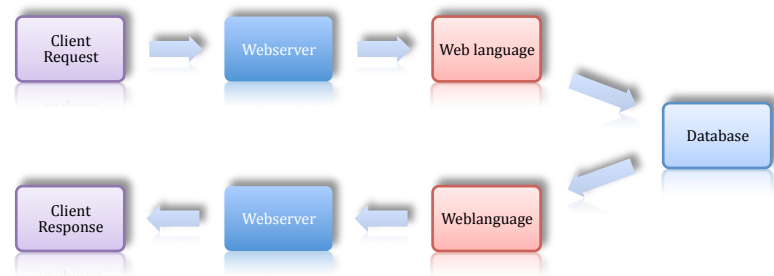


Figure 1: Path from request to response with database calls

According to Kotsis (Kotsis, 2004), PHP is more efficient than Java on a small scale, because its interpreter is compiled into the web-server as a component. This allows for a fast transfer of information between the web-server and the PHP interpreter. Although this is a major advantage for small projects, and low load servers, this is also a huge disadvantage when you want to be able to spread the load over multiple servers. In PHP it is not possible to run the web-server on one computer, and the interpreter on the other, as can be done in Java. If the interpreter or the web-server is the bottleneck, splitting them over two computers can mean a big speed-up, which cannot be done in PHP.

When comparing .Net's C# with Java, as is done by Bjorkqvist (Bjorkqvist, 2004) and Cecchet (Cecchet, Chanda, Elnikety, Zwaenepoel, & Zwaenepoel, 2003), there is not one clear winner, on some benchmarks C# out performs Java, on some other Java wins. It is clear however, that the option to compile code in advance makes both Java and .Net run much faster. This prevents the interpreter from having to interpret each separate request.

When comparing .Net's C# with Java, as is done by Bjorkqvist (Bjorkqvist, 2004) and Cecchet (Cecchet, Chanda, Elnikety, Zwaenepoel, & Zwaenepoel, 2003), there is not one clear winner, on some benchmarks C# out performs Java, on some other Java wins. It is clear however, that the option to compile code in advance makes both Java and .Net run much faster. This prevents the interpreter from having to interpret each separate request.

4.2.1 Experiment

For this research an experiment has been set up, running all three languages on a server with similar specifications using virtual machines. The machine on which the experiments are run is an Asus M2N-E motherboard, with 4gb ram and an AMD X2 5200+ processor. An advantage of running the experiments on one machine using virtual machines is that the delay caused by networks can be reduced.

For a simulation as close as possible to real world setups, one server will be dedicated to databases, and another will be the webserver, implementing one of the three languages. Each virtual machine will have 512mb ram available.

To test the influence of the database in performance, the tests will also be run on Java using SQL server, to see if there is a difference in performance.

4.2.1.1 Testing method

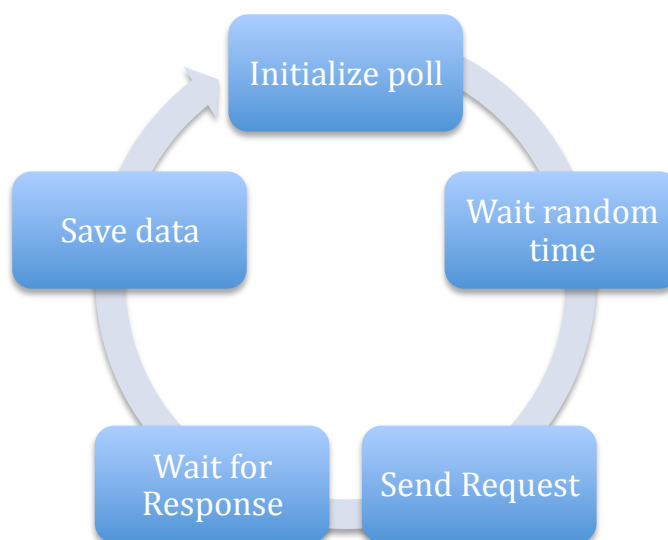
A HTML Ajax page will be used to create a performance testing script. This way, response time can be measured quickly, efficiently and accurate to the millisecond. Another advantage of this approach is that the measurement technique is independent form the language it is testing, and can therefore be used unchanged for each of the languages, making it possible to objectively compare the languages.

For Java and .Net, caching will not be used, enabling testing of actually calculation times. Of course caching could be huge advantage in performance these two languages have over PHP, but it would interfere with the purpose of this test, to compare the calculation times of the three languages.

4.2.1.2 Real world situation simulated

The real world situation we want to simulate is a large number of clients are sending requests to a webserver. Of course, these will never occur at the same time, coming in to the webserver wave after wave. Far more likely is that the time in between requests for each client will differ.

To deal with this issue, a large number of Javascript objects will be generated on the page, each simulating one client. Each of these objects will wait a random time, the ceiling of this will be configured at the start of the test, than fire a HTTP-Request at the specified page. This cycle will be repeated a number of time, so that a measurement under near constant load can be done.



The time it takes for the webserver to respond to the HTTP-Request will be measured, and an average will be calculated over the total of clients executing the requests. This test will be repeated for each languages with different amounts of clients, to see how much they can handle.

4.2.1.3 Tested operations

The test page will be sending requests to a page that selects a large query from a database, and proceeds to do some CPU and memory intensive operations on it, to maximize the difference between the languages.

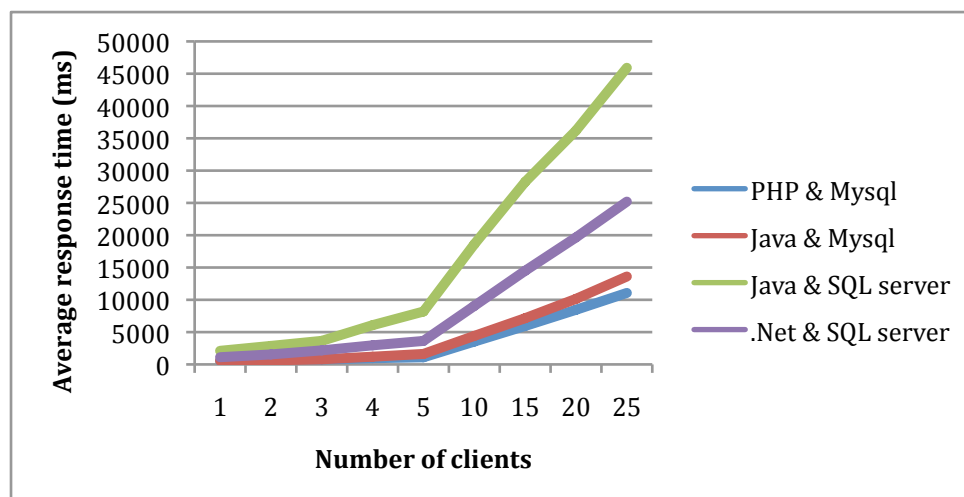
A large set of texts will be retrieved from a database, and stored in a two dimensional array, containing the id, the title and the body of the text. After this is done, the following operations will be done:

- All texts with a title that has a specific substring in it will be copied to a new array.
- All texts with an id that is an odd number will be discarded.
- All texts will be stored in a third array alphabetically, first according to title, than according to text.
- Finally, the text with the longest body is printed.

By executing these four operations, a language is tested in string comparison, integer operations, storage speed and finally printing text, important operations that are almost always required in web applications.

4.2.1.4 Results

The results shown below are from a series of tests, ran on a random filled database with 47318 rows, with a random wait time between 0 and 3000ms. Before being able to complete all the tests, the maximum memory size of Java had to be increased to prevent heap-errors.



4.2.1.5 Conclusion

For large projects, where scalability is important, PHP cannot compete with Java and C# (.Net) because of the lack of support for distributed computing, caching and load balancing. However, looking at smaller projects, hosted on a single server, the fact that PHP is directly tied in with the webserver appears to give it a slight performance edge over Java and .Net.

Which database is used seems to tremendous influence, as the time difference between Java with MySQL and Java with SQL server is larger than the complete calculation time for .Net and SQL server. This means that the performance disadvantage might not be in a slower server, but in the implementation of this server in Java. If the server had been the bottleneck .Net would have been slower aswell.

4.3 Collective ownership

Of course collective ownership is very important in every project, therefore this is also something the three languages are tested on. Testing this, is done by letting a group of test subjects complete two programming exercises, the second on being a revision on part one generated by someone else. How easy it is to enforce collective ownership in a language is than tested by interviewing the test subject, and asking them how easy it was to make the revisions on someone else's code, in comparison to if it would have been their own work.

Because a pretty complete code-style was suggested for the experiments, readability of the code wasn't a problem for any of the test subjects, so there seemed to be a tie.

However, when looking into the languages and the code-conformations used online, it becomes apparent that the .Net framework has a much more standardized code-style. For Java and PHP, the code-style used has to be clearly dictated and written down, whereas for .Net it is clear from the start of the project, which code-style should be used.

4.4 Code re-use

If a programmer has been using a certain language for a while, he often has a large set of previous work from which he can obtain pieces, or snippets, of code. The ease with which this can be done is characteristic for how good a language supports code-reuse. This is tested in the same experiments as mentioned above, but in the interviews the subjects are asked how much of the code they've actually

written from scratch, and how much they managed to copy from other projects or even examples online.

As expected, the subjects used pieces of code from previous projects in all three languages, but the extent to which they did so varied. The code ranged from a simple database connect-function in PHP to an entire database class in .Net. For this part it is clear that the fact that PHP is more a scripting language, and less an Object Oriented Programming (OOP) language than .Net and Java is a huge disadvantage for code re-use.

Of course it is still possible to write OOP code in PHP, but in Java and .Net the user is nudged, or even forced, into this direction, whereas in PHP it is left entirely up to the programmer. This liberty often results in new programmers having an “as long as it works” attitude, and ignoring the OOP developing way. This decision results in code that is usually very specific for each project, which is undesirable from a code re-use perspective.

Another advantage that both Java and .Net have is the ability to compile separate classes into executable files, .dll for .Net and jar's for Java. These files can be included in a new project, without any modifications. This can be very helpful for commonly used objects, such as for instance database classes or user classes.

In the code re-use area Java and .Net are far ahead of PHP, which is especially important for projects where large sections of code could have been re-used.

4.5 Grading

As mentioned above, the score a language gets on a specific property depends on project size. Because this can influence the final outcome, I've decided to create two separate overview table. The first one is for smaller projects, the second for the larger ones.

What is considered a “smaller project” in this research is a simple website, for instance presenting some text form a database, which is editable by an administrator in some way. A “large project” is a project with more intelligence, or a link to large back-end servers or other projects. This could vary from a large management, statistics or accounting suite to navigation software.

“Table 1: Score chart for smaller projects” shows that PHP is the best option for smaller projects, mostly due to its high performance on single servers.

The fact that PHP is lacking a bit in for instance collective ownership, isn't as important as it would be for larger projects.

Table 1: Score chart for smaller projects

| | PHP | Java | .Net |
|----------------------|------------|----------|-------------|
| Deployability | 8 | 6 | 8 |
| Performance | 8 | 6 | 5 |
| Collective Ownership | 7 | 8 | 8 |
| Code re-use | 7 | 8 | 8 |
| Total | 7,5 | 7 | 7,25 |

"Table 2: Score chart for larger projects" shows that PHP can't compete with Java and .Net when looking at larger projects, but there isn't a real difference between Java and .Net. The choice between Java and .Net will most often be made on specific features for each of the different languages, or the existing expertise of programmers.

Table 2: Score chart for larger projects

| | PHP | Java | .Net |
|----------------------|-------------|----------|----------|
| Deployability | 6 | 8 | 8 |
| Performance | 6 | 8 | 8 |
| Collective Ownership | 7 | 8 | 8 |
| Code re-use | 6 | 8 | 8 |
| Total | 6,75 | 8 | 8 |

On the whole, .Net seems to be the winner, being a great candidate for both larger and smaller projects. PHP clearly isn't the best-suited language for larger projects, or projects demanding a great deal of performance. Java is most likely to complex to be worth the effort for smaller projects, but could be a good candidate for large projects, such as for instance big backend services.

5 In practice

For this project two Dutch companies were contacted to see if their experience is in line with the results from the research. The first contacted company is Hyves, the largest social networking website in the Netherlands. The second company is TomTom, Europe's leading navigation software supplier, and one of the largest in the world.

5.1 Hyves

When Hyves first started out as a small company, they started out in PHP because of the low learning curve and low hosting cost. Java and .Net hosting can be very costly, because there aren't a lot of low entry-level hosting packages available. Also, their designers often already knew PHP, making it very suitable for them to start out with.

The first large growth Hyves went through put a strain on PHP, requiring them to work in frameworks, like is done in Java and .Net projects, and use opcode caching. But being this far into the development of a project like Hyves makes starting over in another language very difficult. Still the flexibility of PHP made it very suitable for handling user generated content (because of the array model). Another nice feature is that the time-to-market remains short because of the short development time.

Currently, as Hyves has a large technical staff, huge amounts of source code, and a big server park, the disadvantages of PHP become more pressing. Java and .Net are both compiled languages, with shared memory facilities and database connection pooling. All these features have to be implemented by the programmer in PHP, instead of being able to use off-the-shelf solutions. Creating custom tooling and architectures is a time intensive process.

5.2 TomTom

TomTom has large backend servers, storing and calculating huge amounts of data sent from and to customers. In the past these were run with PHP, but TomTom has made a transfer to Java. Of course, such a fundamental change is always carefully considered and motivated.

The complexity of services being supplied by TomTom required a break with PHP for several reasons. The first and foremost being people, when looking

for a technical staff that can create the complex applications and services TomTom was looking for, you will find that very few of these people work in PHP. Another reason for changing the development platform is the availability of tooling and certain libraries that aren't available for PHP. Especially tools for quality control, such as for instance unit testing, aren't available in PHP.

Also, PHP lacks possibilities for "*Inversion of Control*" and "*Aspect Oriented Programming*" making it a very flat language. Inversion of Control (IoC) inverts the flow of control of a system in comparison to traditional architecture. Instead of defining a sequence of procedures, the user of IoC writes responses to certain events or requests. Aspect Oriented Programming (AOP) is a way to improve modularization by handling problems caused by crosscutting concerns, concerns that cannot be encapsulated into a single entity.

Being a flat language might be ideal in some situations, because of the low learning curve and fast development, it's not suitable for the complicated applications TomTom is aiming for.

TomTom chose to move from PHP to Java, rather than .Net because of the company's preference for Open Source. A reason to prefer Open Source software is to limit dependency to another company (in this case Microsoft). A second reason for choosing Java over .Net is the people-factor mentioned before, most of the programmers skilled in the capabilities TomTom requires work in Java.

5.3 Findings

This research has focussed on four different attributes on which the languages involved have been tested, but the interview with TomTom reveals a fifth attribute, which might be more important in business than the first four combined. If no technicians can be found who are capable of working a certain language, performance, deployability, code re-use and collective ownership are worthless.

6 Conclusion

For the smallest of projects, PHP is a very good language. The simplicity of a scripting languages makes it very easy to learn. Looking at performance, PHP is the fastest of the three languages when run on a single server. However, with scalability in mind, PHP has a disadvantage.

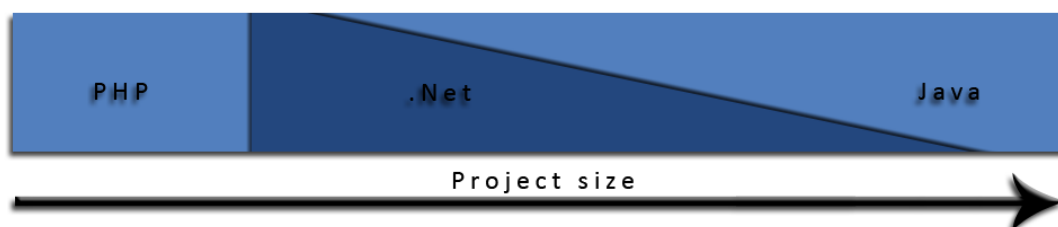
Java and .Net can both out perform PHP on a large number of server because of a better support for load balancing and distributed computing. This is a big advantage for larger projects.

Another advantage Java and .Net have over PHP is the fact that they are much more object orientated. Where the scripting nature of PHP makes it a very easy programming language to learn, code re-use and collective ownership suffer.

In business, another very important part of choosing a programming language is the availability of skilled programmers. The bigger and more complicated applications become, the harder it will be to find programmers. In the current market it is much harder to find good programmers for .Net and PHP projects than for Java projects.

6.1 The best language

In the end project size seems to decide which programming language is best suited. For the smallest of projects, PHP is the best option. For large projects .Net and Java are very evenly matched from a technical point of view. However, the larger a project become, the more important it becomes to be able to find skilled programmers, and in this area Java currently has .Net beat.



7 References

Bjorkqvist, M. (2004). *Object-Oriented Efficiency Comparison: Java, C++ and C#*. <https://www.cs.hut.fi/Opinnot/T-106.290/K2004/Final/mbjorkqv.ps.gz>.

Cecchet, E., Chanda, A., Elnikety, S., Zwaenepoel, J. M., & Zwaenepoel, W. (2003). *Performance Comparison of Middleware Architectures for Generating Dynamic Web Content*. <http://infoscience.epfl.ch/getfile.py?docid=5500&name=middleware2003&format=pdf&version=1>.

Hunter, J., & Crawford, W. (2001). *Java Servlet Programming* (Vol. 2). O'Reilly.

Kotsis, G. (2004). *Performance Comparison of Web-based Database Access*. http://www.tk.uni-linz.ac.at/download/papers/gk_wuxi2002.pdf.

Leong, L. (2003). *Global Internet Offers Big Opportunities for Growth*. Gartner.

Microsoft. (2008). *Windows Server 2008: Web and Application Platform*. Retrieved May 15, 2008 from Microsoft.com: <http://www.microsoft.com/windowsserver2008/en/us/web-application.aspx>

Pitt, L., & Berthon, P. (1999, April). Changing Channels: The Impact of the Internet on Distribution Strategy. *Business Horizons*, 42, pp. 19-29.

Williams, H. E., & Lane, D. (2004). *Web Database Applications with PHP and MySQL*. O'Reilly.

8 Appendix

8.1 Code

8.1.1 Javascript class

```
function Requester(id,maxPolls, wait)
{
  this._id = id;
  this._maxPolls = maxPolls;
  this._maxWaitTime = wait;
  this._currentPoll = 0;
  this._pollsCompleted = 0;
  this._containerDiv = 'requestContainer_'+this._id;
  this._lastResultDiv = 'lastResultDiv_'+this._id;
  this._waitTimesDiv = 'waitTimesDiv_'+this._id;
  this._progressBarDiv = 'progressBarDiv_'+this._id;
  this._waitTimes = [];
  this._avgTime = 0;
  randomWaitTime=Math.floor(Math.random()*maxWaitTime);
  setTimeout("Requesters["+id+"].init()",randomWaitTime);
  //this.init();
}

Requester.prototype._id;

Requester.prototype.getId = function()
{
  return this._id;
}

Requester.prototype.init = function()
{
  document.getElementById(this._containerDiv).innerHTML = "Requester number: " + this._id +
  "<div id='"+this._progressBarDiv+"' class='progressBar'></div>"+
  "<div id='"+this._lastResultDiv+"' class='infoField'>Last result</div>"+
  "<div id='"+this._waitTimesDiv+"' class='infoField'></div><br />";

  this.poll();
}

Requester.prototype.poll = function()
{
  var response = "", startTime, requestId = this._id, currentPoll, randomWaitTime;
  if(this._currentPoll < this._maxPolls)
  {
    executeAjaxCall(requestId,this._currentPoll);
    this._currentPoll++;
  }
}

Requester.prototype.processPoll = function (response, startTime, currentPoll)
{
  var endTime = new Date().getTime();
  var waitTime = endTime-startTime;

  this._waitTimes[currentPoll] = waitTime;
  $(this._lastResultDiv).innerHTML = "Poll returned:<br>["+response+"]<br>";
  $(this._waitTimesDiv).innerHTML = "Last wait time:<br>["+waitTime+"]";

  this._pollsCompleted++;

  var percentageComplete = (this._pollsCompleted/this._maxPolls);
  $(this._progressBarDiv).innerHTML = Math.round(percentageComplete*100)+" %";
  $(this._progressBarDiv).style.width = (percentageComplete*200)+"px";
}
```

```

if(this._pollsCompleted == this._maxPolls || this._currentPoll == this._maxPolls)
  this.calcAvgWaitTime();
}

Requester.prototype.calcAvgWaitTime = function ()
{
  var thisWaitTime = 0, totalWaitTime = 0, numberWaitTime = 0, averageWaitTime = 0;

  for(var i=0; i<this._maxPolls;i++)
  {
    thisWaitTime = this._waitTimes[i];
    if(thisWaitTime>0)
    {
      totalWaitTime += thisWaitTime;
      numberWaitTime++;
    }
  }

  averageWaitTime = totalWaitTime/numberWaitTime;
  $(this._waitTimesDiv).innerHTML += "<br/>Average wait time: ["+averageWaitTime+"]";
  this._avgTime = averageWaitTime;
}

executeAjaxCall = function (requestId, currentPoll)
{
  var startTime = new Date().getTime();
  new Ajax.Request
  (
    target,
    {
      method:'get',
      parameters: {versionForce: startTime},
      onSuccess: function(transport)
      {
        Requesters[requestId].processPoll(transport.responseText, startTime,currentPoll);
      },
      onComplete: function(transport)
      {
        randomWaitTime=Math.floor(Math.random()*maxWaitTime);
        setTimeout("Requesters["+requestId+"].poll()",randomWaitTime);
      }
    },
    requestId,
    startTime,
    currentPoll
  );
}

```

8.1.2 PHP class

```

class DatabaseArray{

    private $texts;
    private $textsMatchingSubstring;
    private $textsEvenIds;
    private $textsSorted;

    private $maxRows = 100000000;
    private $subStringMatch = "aa";

    function __construct()
    {
        mysql_select_db("test",mysql_connect("virtualMySQL", "root", "admin"));
        $sql = "SELECT * FROM text";

        $req = mysql_query($sql);
        $i=0;
        while(($rec = mysql_fetch_array($req)) && $i < $this->maxRows)
        {
            $this->text[$i] = $rec;
            $i ++;
        }
        echo "$i rows found fetched from database <br>";
    } // construct

    public function runTest()
    {
        $output = "";
        $output .= $this->matchSubString();
        $output .= $this->filterOddIds();
        $output .= $this->sortArray();
        $output .= $this->printLongest();
        return $output;
    }

    public function matchSubString()
    {
        $i = 0;
        $iterations = 0;
        foreach($this->texts as $key => $text)
        {
            if(substr_count($text["title"],$this->subStringMatch) > 0)
            {
                $this->textsMatchingSubstring[$i] = $text;
                $i++;
            }
            $iterations++;
        }
        unset($this->texts);
        return "Substring iterations: $iterations <br>";
    }

    public function filterOddIds()
    {
        $i = 0;
        $iterations = 0;
        foreach($this->textsMatchingSubstring as $text)
        {
            $id = $text["id"];
            if(($id/2) == round($id/2))
            {
                $this->textsEvenIds[$i] = $text;
                $i++;
            }
            $iterations++;
        }
        unset($this->textsMatchingSubstring);
    }
}

```

```

    return "Filter iterations: $iterations ($i even)<br>";
}

public function sortArray()
{
    $iterations = 0;
    foreach($this->textsEvenIds as $key => $value)
    {
        $tempLinkTable[$key] = $value["title"];
    }
    natcasesort($tempLinkTable);
    $i = 0;
    foreach($tempLinkTable as $key => $value)
    {
        $this->textsSorted[$i]=$this->textsEvenIds[$key];
        $i++;
        $iterations++;
    }
    unset($this->textsEvenIds);
    return "Sort iterations: $iterations (excl array sort) <br>";
}

public function printLongest()
{
    $longestLength = 0;
    $longestIndex = 0;
    $iterations = 0;
    foreach($this->textsSorted as $key => $text)
    {
        $length=strlen($text["text"]);
        if($length>$longestLength)
        {
            $longestLength = $length;
            $longestIndex = $key;
        }
        $iterations++;
    }
    return "Print longest iterations: $iterations <br>";
}
}

```

8.1.3 Java class

```

public class DatabaseArray {

    private static int idCol = 0;
    private static int titleCol = 1;
    private static int textCol = 2;

    private static int maxRows = 100000;
    private static int colNr = 3;

    private static String subString = "aa";

    private String[][] _texts = new String[maxRows+1][colNr];
    private String[][] _textsMatchingSubString = new String[maxRows+1][colNr];
    private String[][] _textsEvenIds = new String[maxRows+1][colNr];
    private String[][] _textsSorted = new String[maxRows+1][colNr];

    public DatabaseArray()
    {
        this.fillArray();
    }

    public String RunTest()
    {
        String output = "";
        output += matchSubString();
        output += filterOddIds();
        output += sortArray();
        output += printLongest();
        return output;
    }

    private String matchSubString()
    {
        int i = 0, j = 0, iterations = 0;
        while(_texts[i][titleCol] != null)
        {
            if(_texts[i][titleCol].indexOf(subString) != -1)
            {
                _textsMatchingSubString[j] = _texts[i];
                j++;
            }
            i++;
            iterations++;
        }
        return "Substring iterations: "+String.valueOf(iterations)+"<br>";
    }

    private String filterOddIds()
    {
        int i = 0, j = 0, iterations = 0;
        while(_textsMatchingSubString[i][idCol] != null)
        {
            if((Integer.valueOf(_textsMatchingSubString[i][idCol]) % 2) == 0)
            {
                _textsEvenIds[j] = _textsMatchingSubString[i];
                j++;
            }
            i++;
            iterations++;
        }
        return "Filter iterations: "+String.valueOf(iterations)+" (" +j+ " even)<br>";
    }

    private String sortArray()
    {
        int i = 0, outer = 0, inner = 0, run = 0, iterations = 0;
    }

```

```

String first = "", second = "";
while(_textsEventIds[i][titleCol] != null)
{
    _textsSorted[i] = _textsEventIds[i];
    i++;
}
for (outer = 0; outer < (i-run); outer++)
{
    for(inner = 0; inner < (i-1); inner ++)
    {
        if(_textsSorted[inner+1][titleCol] != null)
        {
            first = _textsSorted[inner][titleCol];
            second = _textsSorted[inner+1][titleCol];
            if(first.compareToIgnoreCase(second)>0)
            {
                swap(inner, inner+1);
            }
        }
        iterations++;
    }
    run++;
}
return "Sort iterations: "+String.valueOf(iterations)+"<br>";
}

private void swap(int first, int second)
{
    String[] temp = new String[colNr];
    temp = _textsSorted[first];
    _textsSorted[first] = _textsSorted[second];
    _textsSorted[second] = temp;
}

private String printLongest()
{
    int longestLength = 0, longestId = 0, i = 0, length = 0, iterations = 0;
    while(_textsSorted[i][textCol] != null)
    {
        length = _textsSorted[i][textCol].length();
        if(length > longestLength)
        {
            longestLength = length;
            longestId = i;
        }

        i++;
        iterations++;
    }
    return "print longest iterations: " + String.valueOf(iterations) + "<br>";
}

private String fillArray()
{
    String outPut = "";
    int i=0;
    try
    {
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            outPut += "class not found";
            e.printStackTrace();
        }

        Connection conn = DriverManager.getConnection("jdbc:mysql://192.168.1.78/test", "root", "admin");
        Statement statement = conn.createStatement();
        ResultSet rs = statement.executeQuery("select * from Text");
        while (rs.next()&&i<maxRows)
        {
            _texts[i][idCol]= rs.getString("id");

```

```
        _texts[i][titleCol]= rs.getString("title");
        _texts[i][textCol]= rs.getString("title");
        i++;
    }
}
catch (SQLException e)
{
    outPut += "sql state"+e.getSQLState();
    outPut += "sql except"+e.getMessage();
    e.printStackTrace();
}
return outPut;
}
}
```

8.1.4 C# Class

```

public class DatabaseArray
{
    private bool _limitSQL = false;

    private static int _maxRows = 100000;

    private String _substring = "aa";

    private string _ConnectionString = "Data Source=VIRTUALEXPRESS\\SQLEXPRESS;Initial Catalog=test;Persist Security
Info=True;User ID=sa;Password=sa";

    private List<Record> _texts = new List<Record>();
    private List<Record> _textsMatchingSubstring = new List<Record>();
    private List<Record> _textsEvenIds = new List<Record>();
    private List<Record> _textsSorted = new List<Record>();

    public DatabaseArray()
    {
        fillArray();
    }

    private void fillArray()
    {
        string Query = "SELECT * FROM text";
        if (_limitSQL)
            Query += " WHERE id < " + _maxRows;

        SqlConnection sqlConn = new SqlConnection(_ConnectionString);
        SqlDataAdapter sqlAdapter = new SqlDataAdapter(Query, sqlConn);

        DataSet ds = new DataSet();
        sqlAdapter.Fill(ds);

        foreach (DataRow R in ds.Tables[0].Rows)
        {
            _texts.Add(new Record(R));
        }
    }

    private String matchSubString()
    {
        int iterations = 0;
        foreach (Record R in _texts)
        {
            if (R.Title.IndexOf(_substring) != -1)
            {
                _textsMatchingSubstring.Add(R);
            }
            iterations++;
        }
        return "Substring iterations: " + iterations.ToString() + "<br>";
    }

    private String filterOddIds()
    {
        int iterations = 0;
        foreach (Record R in _textsMatchingSubstring)
        {
            if (R.Id % 2 == 0)
                _textsEvenIds.Add(R);
            iterations++;
        }
        return "Filter iterations: " + iterations.ToString() + " <br>";
    }

    private String sortArray()
    {
        int iterations = 0;
    }

```

```

foreach (Record R in _textsEvenIds)
{
    _textsSorted.Add(R);
}
int run = 1;
int numberOfRows = _textsSorted.Count;
Record temp;
for (int outer = 0; outer < numberOfRows; outer++)
{
    for (int inner = 0; inner < (numberOfRows - run); inner++)
    {
        if (_textsSorted[inner].Title.CompareTo(_textsSorted[inner + 1].Title) > 0)
        {
            temp = _textsSorted[inner];
            _textsSorted[inner] = _textsSorted[inner + 1];
            _textsSorted[inner + 1] = temp;
        }
        iterations++;
    }
    run++;
}
return "Sort iterations: " + iterations.ToString() + "<br>";
}

private String printLongest()
{
    int iterations = 0;
    int longestLength = 0, longestId = 0, i = 0, length = 0;
    foreach (Record R in _textsSorted)
    {
        length = R.Text.Length;
        if (length > longestLength)
        {
            longestLength = length;
            longestId = i;
        }
        i++;
        iterations++;
    }
    return "Print longest iterations: " + iterations.ToString() + "<br>" + "Longest: " + _textsSorted[longestId].Text;
}

public String RunTest()
{
    String output = "";
    output += matchSubString();
    output += filterOddIds();
    output += sortArray();
    output += printLongest();
    return output;
}
}

```