Programming languages in BotBall

Michael Ebenstein
Informationstechnologie (IT)
TGM (Technologisches GewerbeMuseum)
Vienna, Austria
mebenstein@student.tgm.ac.at

Peter Fuchs
Informationstechnologie (IT)
TGM (Technologisches GewerbeMuseum)
Vienna, Austria
pfuchs@student.tgm.ac.at

Keywords—programming; languages; botball; update; suggestion

Programming languages in BotBall

Introduction

BotBall has been around since 1997. 20 years later there has not been any change in the matter of programming languages. All controllers are programmed with C by default, which is a great language, but it is obsolescent. The only possibility for participants to use other languages is to integrate them themselves or, if they take part in Open-BotBall, to use a different controller. Since the target group of BotBall is 10 to 18-year-old pupils it is neither unbearable for the younger participants to do this by themselves nor to pay for a different controller.
This Paper will show the advantages of other languages and their use in BotBall. Note that this paper will only cover the 4 languages Python, C, C++ and Java.

Performance

Performance may seem not so important at first, since most BotBall controllers have more than enough processing power for such “trivial” task used in BotBall, but when it comes to image processing, which is often used in the competitions, performance plays an important role. Furthermore, BotBall must enable participants the possibility to use modern approaches, such as machine learning. Since the field of artificial intelligence and especially machine learning is steadily growing, BotBall must consider that more advanced participants may want to use these methods.

For the performance comparison, we refer to a comparison on raid6.com. The graphs are shown in Fig. 1 and Fig. 2, where the memory usage is shown

C is in this matter one of the most powerful languages. But it simply leaks the languages features and most of all important paradigms, such as object oriented programming. Because ofthis, C++ is the logical choice. Not only does C++ feature a variety of features, it also is as fast as C.

Speaking of machine learning, Python cannot be ignored. It is widely used in this matter, but since it is an interpreted language and not compiled, it is by far not as performant as C or C++.

Java is even slower than Python. It is compiled, but it goes through a lot of processes before that to enable the characteristic platform independence.

Extendibility

Extendibility is very important for various reasons. Even more important, it enables solving problems with many different solutions. Especially for young participants, it would be helpful if they had tools which helped them to get started faster.

Python is probably the most extendible language out there. It is extremely easy to extend features with tools, such as PIP (Python Package Index). Furthermore, Python also features a variety of tools. Java is probably close to Python since all you must do is download the right .jar files and link them, but unfortunately Java does not feature many tools when compared to Python.

C on the other hand is very unpleasant to extend. The compilation of the libraries and the linking of the files is unbearable for the target group. C++ is almost the same as C in this matter. The only difference is that there are way more C++ libraries than for C. Furthermore, there are also lots of Head-Only-libraries.

Expressive/ Readability

Expressive and readable code is important for teamwork and for understanding, what happens. It also helps when learning a language. This would be very useful for beginners.
Readability is in some ways subjective, but we will try to keep it as objective as possible.

C is by far the worst from these languages when it comes to readability or expressiveness. C++ is better, since there has been a lot of work on expressive coding in the last years. The recently published C++17 has brought even more tools to the std. (standard library), to make the code more expressive.

Java has a better readability than C++ because of its documentation which makes it a lot easier for beginners to learn this language. Its expressiveness is nearly the same as C++, but what makes it better is its higher consistency.

Python is the best in this comparison. The main difference is that python doesn't use braces, instead it uses intends which makes it more clear and compact.

Freedom

Freedom in this context means, what possibilities and restrictions a language has. This is important for performance and system design reasons.

C offers probably the most freedom, just after assembly. It´s pointer arithmetic allows fast calculation and complex system design, which would otherwise not be possible.

C++ is pretty much the same as C.

Java doesn't have anything comparable to pointers and has lots of restrictions, such as Boolean not being numeric and so on.

Python as a scripting language has the least freedom to offer.

Language Features

Language features are considered things, such as a garbage collector or safe pointers and so on. These things are a big support for beginners, since they do not have to care about things they don't know about or they don't want to handle.

Python has lots of features and useful build in functionality.

Java has similar features as Python.

C++ offers less of the high-level features.

C offers by far the fewest features and this makes it hard to get started with the language.

The conclusions were drawn from
http://www.jvoegele.com/software/langcomp.html

Language Tools

Language tools refer to the number of existing libraries and frameworks supporting a language.

Many libraries and frameworks support Python. Because of the extendibility of it, it is easy for programmers to support Python.
C is supported by nearly all mayor libraries. Because of this, the libraries are often built simple and without object orientation.

C++ can use all libraries and tools C is able to, and there are even more libraries C++ can use that C isn't able to. Many Frameworks support C++ and most of them are built with it.

Java is not well supported by most libraries and frameworks. There are some specific for Java but these are the minority.

Helpfulness in the job

Working with robots takes a lot of time. For the participants, it would surely be nice if all the work they put into these robots would also bring them some benefits for the future. To enable the best starting conditions for their carrier, it is not only important to gain experience, but also to work with the tools, in this case languages that are used today.

Python is widely used in various areas of programming. Java is also often used, but mostly in the field of databases and communication.

C is still used today, but most companies are switching to more modern languages.

C++ on the other hand is used in nearly every area of computation. Furthermore, it is the standard language of nearly every framework or library.

Out of the box

Out of the box describes how much you must do to get started with a language, consider compilers, IDEs, libraries.

Python is straight forward and easy to use. It comes with an IDE and with the PIP which enables the simple extension of Python.

Java is similar, if you use something like eclipse, but the linking can be more difficult than in Python.

C++ is more difficult considering the existing libraries. It does have the most important tools and containers, but doesn't come with support for graphic related stuff, but this is insignificant for BotBall.

C is like C++ but offers even less support for modern libraries or frameworks.

Compatibility with existing systems

Compatibility with existing systems plays a major role, since nobody want to make a new system from scratch just for language support.

Python is rather easy to implement. It can simply use the existing C functions. A controller for BotBall with Python support has already been created by the TGM in Vienna, named Hedgehog controller. This shows that there shouldn’t be a problem with Python.
C++ can also simply use the C functions, but the compilers must be changed.

Java is more difficult, but it also has the functionality for calling C functions.

Conclusion

We made a table showing the results, but did not weight the factors, since we want to keep it as objective as possible.

	Python	C	C++	Java
Performance	2.5	1	1	3
Extendibility	1	2.2	2	1.5
Expressive/ Readability	1	3	2.5	2
Freedom	2.5	1	1	2.5
Language Features	1	3	2.2	1
Language Tools	1.8	1.5	1	3
Helpfulness in the job	1.5	1.8	1	1.8
Out of the box	1	2.5	2	1.2
Compatibility with existing systems	1.5	1	1.1	2.3
Summary	***14.8***	***17***	***13.8***	***18.3***

This table shows that C++ clearly should be used, closely followed by Python. Java and the currently used language C are further behind.

Therefore, we would suggest KISS to implement C++ and Python support by default on their controllers.

Line size Kb	**Python**	**C++ (g++)**	**C (gcc)**	**Python3**	**Java (openJDK)**	**Java (Sun)**	**Java (gcj)**
256	7	7	2	17	39	38	451
512	32	26	8	81	162	157	1783
768	78	60	19	201	381	371	3937
1024	144	107	34	373	711	696	6952
1280	232	167	53	598	1161	1145	10744
1536	342	242	76	877	1751	1739	15372
1792	476	329	104	1211	2489	2478	20819
2048	634	431	136	1598	3370	3358	27132
2304	815	546	173	2039	4453	4448	34302
2560	1019	675	214	2533	5710	5719	42330
2816	1248	817	259	3070	7146	7186	51118
3072	1497	972	309	3659	8852	8983	60779
3328	1771	1142	363	4300	10784	10916	71275
3584	2064	1324	423	4992	12696	12867	82619
3840	2381	1522	487	5729	14861	15053	94686
4096	2720	1731	555	6534	17262	17426	107887

Fig. 1 Speed comparison of the languages. The table shows number of seconds taken to complete every testing stage.

Line size Kb	***C (gcc)***	***C++ (G++)***	***Python***	***Python3***	***Java (gcj)***	***Java (OpenJDK)***	***Java (Sun)***
0	1,668	2,932	5,352	10,328	49,156	72,4832	658,560
256	1,928	3,444	6,384	13,404	68,320	725,852	661,056
512	2,184	3,956	5,876	16,476	76,200	725,852	661,056
768	2,440	3,956	7,676	19,548	84,392	725,852	661,056
1024	2,696	4,980	6,388	14,420	92,584	725,852	661,056
1280	2,952	4,980	9,212	15,444	110,072	725,852	661,056
1536	3,208	4,980	6,900	16,468	118,264	725,852	662,080
1792	3,464	4,980	7,156	17,492	126,976	725,852	662,080
2048	3,720	7,028	11,516	18,516	126,976	725,852	662,080
2304	3,976	7,028	7,668	19,540	161,824	725,852	662,080
2560	4,232	7,028	7,924	20,564	161,824	725,852	662,080
2816	4,488	7,028	8,180	21,588	161,824	725,852	662,080
3072	4,744	7,028	8,436	22,612	161,824	725,852	662,080
3328	5,000	7,028	8,692	23,636	170,536	725,852	662,080
3584	5,256	7,028	12,536	24,660	170,536	725,852	662,080
3840	5,512	7,028	9,204	25,684	170,536	725,852	662,080
4096	5,768	11,124	9,460	26,708	170,536	725,852	662,080

Fig.2 Memory usage comparison. Numbers shown are KiB used in every testing stage.