Java auto-porting

The – biggest problem of of CsPorter is that it doesn't understand what package is equivalent of C#'s namespace. Commonly it adds to the namespace from using statement “**com.aspose.ms.**” and in the result we have somethig like this: **import com.aspose.ms.System.Collections.Generic** instead of **import java.util.\*** for example.

As for strings – there is another problem. Porter doesn't understand default string operations such as +=. May be this problem can be solved with StringBuilder, But I don't know for sure.

CsPorter tries to add a new import called StringExtensions with operations for string, but there is now such package for java as such as for c#.

Big advantage of porter is that it can translate c# routine code such as properties with get and set methods to java. Also it can translate usual operations, but as for specific operations with different relations to other library projects – there could be problems. This problem rises because CsPorter doesn't port whole project, it ports only file which project includes.

While having the porting process I had no problems with porting interfaces, but porting of porting core functions was quite problematic, because of problems shown earlier.

The process of porting is unusual. At first we have to upload .csproj file to the program, after that we have to update data from this file, after that we will see full .xml file with all classes which are ready to be ported. Here we have a small problem. CsPorter wants to port Assemblyinfo.cs file. To solve this we have to modify created xml file and delete this file from the list of file to port. Also we have to specify a folder where we store ported files. And after that we can select option “Convert”

**Conclusion:**

Porting with CsPorter helps to convert boring part of the code such as interfaces and properties with their getting and setting methods. But as for functions – we have to look through all code and rewrite all imports and adapt some C# specific libraries and codestyle to Java code Style. Quantity of changes depends on difficulty of code and quantity of platform specific methods. As for generics – I don't know. I have not tested this yet