Design Document for JavaToUML project

Wei Yao

1. Project Goal and Requirements
   1. Agile engineering process: Kanban style

A weekly Kanban board is maintained to monitor project progress. <https://waffle.io/Wei2015/JavaToUML>

* 1. UML Diagrams requirements
* Dependencies & Uses Relationships for Interfaces Only
* Class Declarations with optional Extends and Implements
* Only Include Public Methods (ignore private, package and protected scope)
* Only Include Private and Public Attributes (ignore package and protected scope)
* Java Setters and Getters:  Must Support also Java Style Public Attributes as "setters and getters"
* Must Include Types for Attributes, Parameters and Return types on Methods
* Classifier vs Attributes Compartment:

1. Tools and API used in the project
   1. Java parse to String API: JavaParser <http://javaparser.org/>

I have used JavaParser to parse java code in my first part of project. JavaParser can get an Abstract Syntax Tree (AST) from Java code. By using this data structure, all components in a piece of Java code can be retrieved, like attribute declaration name, type, method name, return type and so on.

The Javadoc of JavaParser can be found here. <http://www.javadoc.io/doc/com.github.javaparser/javaparser-core/3.1.3>

Some helpful information on how to use JavaParser can be found here: <https://github.com/javaparser/javaparser/wiki/Manual>

To used JavaParser, I have downloaded the javaparser-core-3.1.0-beta.2.jar and and to my Intellj IDE for code implementation.

* 1. String parse to UML diagram API: plantUML <http://plantuml.com>

I have chosen plantUML tool to implement my output to UML diagram. I have downloaded plantuml.jar and added it to my Intellj IDE for code implementation.

All string inputs for plantuml must start with @startuml and end with @enduml. I choose output image format is SVG, which is supported by all modern browsers and can be zoomed in and out without losing resolution.