Visualizing Code Annotations Distribution

FREE AND CLARIFIED CONSENT TERM (**FCCT**)

You are being invited to participate in the research entitled "Visualizing Code Annotations Distribution". This study aims to measure how our proposed visualization approach, implemented in the tool AVisualizer, is able to effectively display information about code annotations being used in java software systems.

Participation: The experiment is divided in fourt parts. In the first you will be asked a series of objective questions about yourself. Then, in the second part, you will be asked to watch a 17 minute video that presents the AVisualizer and how it works. Following, in the third part, you will be asked 10 objective questions that you must use the tool to answer. Finally, in the fourth part, you will be prompted with a series of objectives questions about your impressions and opinion about the tool. We estimate the experiment takes between 40-50 minutes.

Confidentiality and anonymity: The information in this research is strictly confidential, and disclosed only in scientific events or publications and will only be used for this purpose, with no identification of the participant(s), unless between those responsible for the study, ensuring full confidentiality about their participation.

Contact information for those responsible for the research: During the research period, you have the right to ask any questions or ask for any other clarification, simply contacting the principal researcher by the email phyllipe@inatel.br

Voluntary Participation: You have the right to refuse to participate in the referred survey or to withdraw from this study, at any time, without prejudice or retaliation, for your voluntary decision. To quit, just leave the site before completing the survey.

Data retention: The data will be kept after the completion of the project for 5 years.

Consent Registration: Once the questionnaire is being carried out using an electronic form, you must check the option in which you affirm that you agree with the participation of the study and make a copy of it after completing it. You can also request a copy of the document signed by the researchers.

*Obrigatório

1. Informed consent * *

Marcar apenas uma oval.

- ◯ I consent to participate in this survey. *Pular para a pergunta 2*
- ◯ I do not consent, and hence prefer not to participate
  Pular para a seção 7 (We understand you do not wish to participate in the study! :))

Personal Information

In this section you will be asked questions about yourself, experience with software development and code annotations

2. Name *

_____

3. Email *

_____

4. Current Role *

Marcar apenas uma oval.

- ◯ Undergraduate Student (Computer Engineering, Computer Science, or related areas)
- ◯ Graduate Student (Masters - Computer Engineering, Computer Science, or related areas)
- ◯ Graduate Student (PhD - Computer Engineering, Computer Science, or related areas)
- ◯ Professional Software Developer
- ◯ Researcher
- ◯ Outro: _____

5. How well are you familiar with Code Annotations? (Other languages contain similar feature, such as attributes (C#) and decorators (Python). When answering about your experience with code annotations, also consider your familiarity with other similar features.) *

Marcar apenas uma oval.

	1	2	3	4	5	
Not familiar	◯	◯	◯	◯	◯	Very Familiar

6. What is your primary programming language? *

Marcar apenas uma oval.

- ◯ Java
- ◯ C#
- ◯ C++
- ◯ Python
- ◯ JavaScript
- ◯ Typescript
- ◯ C
- ◯ Outro: _____

Pular para a seção 3 (Code Annotations - Brief Explanation)

Code Annotations - Brief Explanation

Code annotations, or simply annotations, are a feature of the Java language. They are used to configure custom metadata on programming elements (methods, fields, etc)

As an example, we have the @Test annotation from the JUnit framework. Consider the code snippet below:

```
@Test
public void testExample(){
    assertTrue(true);
}
```

The annotation @Test is configuring the method "testExample" and will be consumed by the JUnit framework.

Annotations can also contain arguments. Consider the example below:

```
@Annotation1
@Annotation2(value = 13)
private int field1;
```

Both annotations @Annotation1 and @Annotation2 are configuring the field "field1". The @Annotation1 has 0 (zero) arguments. The @Annotation2 has 1 (one) argument.

Important Definitions
- Sometimes we can use the word "annotation" or "code annotation". They are equivalent
- A group of related annotations we refer to as "annotation schema" or simply "schema". For instance we have the "org.junit" schema and "javax.persistence" schema

Pular para a seção 4 (AVisualizer - Tutorial Section)

AVisualizer - Tutorial Section

The AVisualizer is an open-source polymetric software visualization tool. It implements our approach to display code annotations usage/distribution in a target java software. We use a circle packing strategy to display the system.

The tool has three views that you can navigate to and from.

System View: Displays java packages (circles with dashed lines on the border) and code annotations schemas (colored circles) per package. The size of colored circle is proportional to the number of annotations of that schema inside the package. The color of these circle are related to their schema

Package View: Displays classes (white circles) inside the packages. Inside each class there are colored circles that represents annotations being used by the classes. The color is also related to the schema. The size of these colored circles is defined by the LOCAD (Lines of Code in Annotation Declaration) metric. In other words, LOCAD, is the number of lines used to write the annotation. To access the Package View, simply click on any annotation schema from the System View. To go back to the System View, click on the grayish-background in the external part.

- Example: The @Override has LOCAD = 1

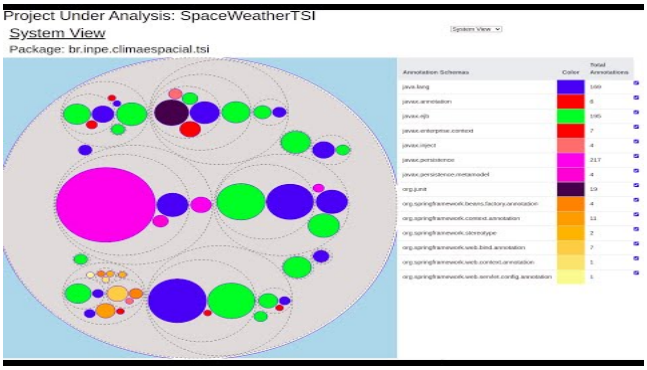
Class View: Display classes (still white circles). But now you can see how the annotations (still colored circles) are grouped inside a class. In other words, we can see how many annotations are configuring any specific method, or class member. Grouped annotations will have a circle frame containing them. Annotations without a frame means they are configuring the class itself. The size of annotations (colored circles) is defined by the AA (Arguments in Annotations) metric. That is, the size is defined by the number of arguments/attributes passed as parameter when writing the annotation. To access the Class View, you must be on the Package View and then simply click on any annotation. To go back just click the exterior grayish-background. Notice that we switch metric from the Package View (uses LOCAD) to the Class View (uses AA)

- Example: The annotation @Column(name = "Id") has AA = 1 and LOCAD = 1
- Example: The annotation @Override has AA = 0 and LOCAD = 1

Following is the Tutorial Video.
Please watch the video before proceeding. **It is 17 minutes long**

The link of the video will be available to you. You can re-watch as many times as you wish during the experiment

AVisualizer Tutorial - Video



http://youtube.com/watch?v=oodiUHM1qBc

Pular para a pergunta 7

Using the AVisualizer

In this section, you will be prompted with 10 objective questions regarding code annotations.

To answer these questions you should have the AVisulizer tool open (in another Tab or window).

You can access it on the link: https://avisualizer.herokuapp.com/

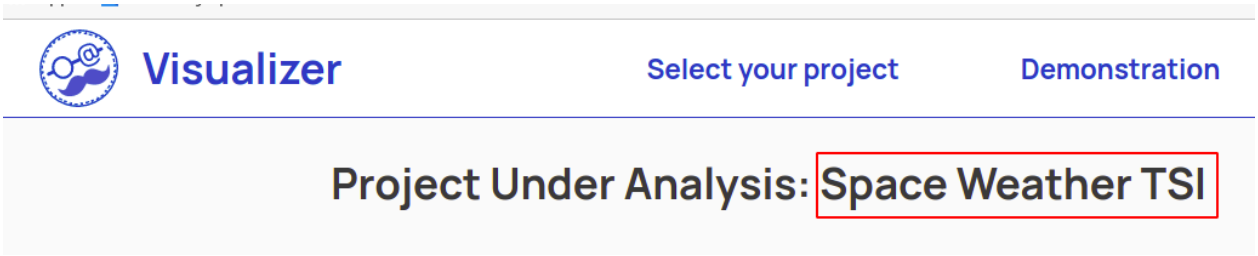
When the AVisuliazer web application loads, verify that the Project Under Analysis is "SpaceWeatherTSI". This information is at the top left of the page. Adjust the zoom on your web browser to best suit you. If all circles are black, please refresh the page (F5)

The AVisuliazer may take 2-3 min to load.

After the AVisualizer is loaded you may proceed

AVisualizer Link: https://avisualizer.herokuapp.com/

Certify the "Project Under Analysis" is the "SpaceWeatherTSI", shown in the image below



AVisualizer Tutorial

You can rewatch the tutorial at any time on the following link: https://youtu.be/oodiUHM1qBc

Objective Questions

Following is a list of 10 objective questions

7. What annotation schema is located in a single package? *

Marcar apenas uma oval.

- java.lang
- javax.ejb
- org.junit
- javax.persistence
- javax.annotation

8. What annotation schema is located in more packages ? (more distributed) *

Marcar apenas uma oval.

- javax.persistence
- org.junit
- java.lang
- javax.inject
- javax.annotation

9. Which annotation schema contains the largest amount of annotations being used? *

Marcar apenas uma oval.

- org.junit
- org.springframework.stereotype
- java.lang
- javax.ejb
- javax.persistence

10. What class contains the highest number of javax.persistence annotations? *

Marcar apenas uma oval.

- TsiHDU
- FitsDownload
- TsiData
- TsiFits
- TsiRingsArea

11. What package contains classes being mapped to databases (usage of javax.persistence)? *

Marcar apenas uma oval.

- br.inpe.climaespacial.tsi.business
- br.inpe.climaespacial.tsi.collector
- br.inpe.climaespacial.tsi.entity.model
- br.inpe.climaespacial.tsi.collector.scheduler
- br.inpe.climaespacial.tsi.business.impl

12. What package is mostly concerned with web controllers (usage of org.springframework.web.bind.annotation)? *

Marcar apenas uma oval.

- br.inpe.climaespacial.tsi.business
- br.inpe.climaespacial.tsi.collector
- br.inpe.climaespacial.tsi.entity.model
- br.inpe.climaespacial.tsi.viewer.rest
- br.inpe.climaespacial.tsi.business.impl

13. How many packages contains unit testing class(es) *

Marcar apenas uma oval.

- None
- Only 1
- From 2 to 3
- From 3 to 5
- More than 5

14. What javax.persistence annotation has the highest LOCAD (lines of code per annotation) value? *

Marcar apenas uma oval.

- Column
- Table
- GeneratedValue
- OneToOne
- Entity

15. In the class br.inpe.climaespacial.tsi.entity.model.TsiHDU (fully-qualified name), what code element (method, field, class definition, etc..) has more annotations configuring it? *

Marcar apenas uma oval.

- Field - id
- Method - getId
- Method - setId
- Field - hduType
- TsiHDU - Class Definition

16. What annotation from the org.springframework.web.bind.annotation schema contains more attributes/arguments (Annotation Attribute metric - AA) *

Marcar apenas uma oval.

- RestController
- RequestMapping
- Bean
- Autowired
- RequestScope

Pular para a pergunta 17

Impressions of the AVisualizer

In this section we are interested in your opinions and impressions of the tool.

The AVisualizer is still in its initial development, and the main idea was to demonstrate the approach we are developing to visualize code annotations in a target software system.

For each statement below, check 1 for strongly disagree, 2 for disagree, 3 for neither disagree nor agree, 4 for agree or 5 for strongly agree.

17. Learning how to use the AVisualizer was easy to me *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

18. I would easily detect all annotation schemas that is being used in a java system using the AVisualizer tool. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

19. I would easily detect all annotation schemas that is being used in a java system simply inspecting the code. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

20. I can easily navigate to and from the packages and classes being analyzed with the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

21. I can easily see how code annotations are distributed in the system under analysis using the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

22. I can easily identify what java package I am currently inspecting using the AVisualizer *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

23. I can easily see how many annotation schemas are being used inside a java package using the AVisualizer *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

24. I can easily see how many annotation schemas are being used inside a java class using the AVisualizer *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

25. I can easily identify the class I'm inspecting in the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

26. I can easily identify classes highly coupled to annotation schemas using the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

27. I can easily identify classes highly coupled to annotation schemas simply inspecting the code *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

28. I can easily spot potential misplaced code annotations using the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

29. I can easily spot potential misplaced code annotations simply inspecting the code *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

30. I can easily spot large annotations using the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

31. I can easily spot large annotations simply inspecting the code. *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

32. I can easily identify java packages with different responsibilities using the AVisualizer tool *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

33. If I were a newcomer developer on a team, I would you use the tool to get a general view of the project I am working on. (Consider that your project uses code annotations) *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

34. If the tool was a plugin for an IDE (Eclipse, Visual Studio, Intellij, etc), I would use it to analyze the project I am working on *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

35. If the tool was available as a web application, I would use it to analyze the project I am working on *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

36. From what was presented, I consider the AVisualizer tool easy to use, easy to detect code annotations being used and easy to navigate (between the different views) *

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

37. What role do you feel the AVisualizer tool is more useful? (Check all that applies) *

Marque todas que se aplicam.

- [ ] Developer
- [ ] Architect
- [ ] Framework Developer
- [ ] Tester
- [ ] QA (Quality Assurance)
- [ ] Project Manager
- [ ] Newcomer developer (just joined a team, but has previous experience)

Outro: ☐ _____

38. In which of the scenarios below would you use the AVisualizer tool? (Consider that you belong to a team that is currently working on a java system. And you have the AVisualizer tool available) *

Marcar apenas uma oval.

- ◯ To check for misplaced annotations, or very large annotations.
- ◯ To detect java packages with certain responsibilities before I start adding my own features to the code
- ◯ To see every annotation schema being used, and then study these annotations and they are used for
- ◯ Outro: _____

39. Describe your overall evaluation of the AVisualizer tool *

_____

_____

_____

_____

_____

We understand you do not wish to participate in the study! :)