Design and Documentation Change Log

All directories listed here refer to directories in our drive documentation folder.
This document was last updated 6/9/16

Final Release Changes

The following design changes have been made since the last release:

@MinLen

Many rules were added to more fully flesh out how this annotation type interacts with other annotations. See Index Checker Rules for more details.

The following documents have been added or changed, independent of the changes mentioned above

Extensibility Exercise

A new document for explaining our projects scalability towards implementing support for multiple arrays.
Extensibility Exercise.

Postmortem/Updated Timeline

New documents showing how our schedules actually worked out, and reviews on unexpected turns and problems.
Postmortem. TimeLine.

Final Draft Release Changes

The following design changes have been made since the last release:

@MinLen

Added a new annotation type that describes the minimum length of arrays. Details in the new user manual (see user manual change below)

More Intro/Transfer rules

We have added more intro and transfer rules to help eliminate false warnings. These are specifically the Math.min() call and using and (int + 1) to create an array. Details in the Index_Checker_Rules document.

The following documents have been added or changed, independent of the changes mentioned above

User Manual

The Majority of this document has been converted into LaTeX and added as a chapter to the checker framework's user manual. It has been adapted to fit in as a chapter, and not a standalone document. The user manual in this drive folder will no longer be maintained and removed after we are sure nothing more needs to be copied over.

Index_Checker_Rules

The Transfer and Dataflows rules document has been changed to Index_Checker_rules to encapsulate all of the rules our checker currently follows (documentation/Design spec).

Design patterns used

Moved to its own document in documentation/Design spec

Documentation Change Log (this document)

Separated from Design patterns used document

Extensibility Excersize

New document detailing the difficulties in further expanding our project. It is currently in a draft state, as it is due Monday, June 6th

Release-candidate Changes

User distribution

We now are drafting releases via github and creating user distributions (pre-compiled builds) for users. The files can be obtained via download on our release page.

Code Coverage

Added in developer manual

String Support

The checker now also checks calls on <some string>.charAt(i) to see if i is in the bounds of the string. The documentation has not been updated to cover this change.

The following documents have been added or changed, independent of the changes mentioned above

User Manual

Once again, much of the User manual has been re-organized and re-worded to become clearer and more precise.

Code Review

New! Can be found in folder User Studies and Code Review

User Tests

New! Can be found in folder User Studies and Code Review

Weekly Plan

Added an accomplished tab so progress is made public

ZX-ing case study

New case study!

Feature-complete Release Changes

Between 5/17/16 and 5/20/16, The following designs have changed in the following ways:

We have abandoned the idea of having a type annotation having information about more than one array/list. In other words, each type annotation can only contain one value, and consequently only have legal access to one array/list at a time. This design choice was made because of the large risk of not completing it in time. The User Manual and design spec/Transfer Dataflow rules have been updated to reflect this change.

This change lead to the necessity of special case considerations for -1, as it will no longer be introduced as a @IndexOrLow("*"). Changes have been made in design spec/Transfer Dataflow rules to reflect this.

List usage has been fully documented in The User Manual and implemented, design spec/Transfer Dataflow rules, and our test code documentation has been updated. We have decided to forgo Iterator support for now and possibly reconsider if it proves to be an important feature.

The following documents have been changed, independent of the changes mentioned above

Developer Manual

More test information has been added, specifically where to put new tests and how to enable Travis CI for forked repositories

User Manual

The walkthrough example has been significantly simplified and moved to before the detailed type details in order to familiarize the user with what the checker is doing.

Beta Release Changes

From between 5/2/16 and 5/17/16, The following designs/documents have changed in the following way:

Manuals/User Manual

A Large portion of this has been redone. The entire walkthrough example is new. The image explaining our types has been replaced.

Manuals/Developer Manual

Much of the copied information from the checker-framework manual has been removed. Now the obtaining source/build section simply points to the building section in the checker-framework manual with notes on the few things that need to be done differently. The downside was exchanging having one streamlined process with a slightly harder process. The upside that outweighed this is the elimination of the possibility of having conflicting build instructions if the checker-framework's build instructions changed without ours.

Design spec/Transfer Dataflow rules

A new section has been added titled Rules for side-effecting/re-assignment near the end of the document. This is the section that really differentiates the difference between lists and arrays. For now the functionality described is way too complex, for now we are just trying to implement losing information whenever side effecting occurs. This design decision was made due to time concerns.

Design spec/Development process

Overall, the design of our project's layout has not changed very much, given that we have to adhere pretty closely to the design the checker-framework sets up for us. One thing that has changed is the features we are supporting:

- The Index Checker will not significantly increase the amount of time it takes to type-check code (new)
- Warns users about the most common IndexOutOfBoundsExceptions that may occur at compile time (previously stated a guarantee we could not support)

Changed stretch goals:

- Create support for iterators that are created for lists
- Compatibility with using a single integer as an index for multiple arrays (currently any index is only allowed to access one array at a time)

Aiming for fast type-checking time was not a big design change, but rather something we just overlooked earlier. If our tool takes too long to run, people simply wouldn't want to use it. Before we were making a guarantee about not getting IndexOutOfBounds errors. This is way to large of a goal and needed to be reduced.

As we worked along on our project, it became clearer which features would be more difficult to implement than others. This is why we removed part of the indexing multiple arrays support to a stretch goal, as well as iterator support for lists.

Uses and Studies/Evaluation Case Studies

Slightly changed evaluation criteria to include post-annotation warnings produced, true errors found, and separated warning that could be suppressed and false errors that needed to be re-wrote despite being correct

Uses and Studies/Case Study: GArrayStack/...

New files