Introduction

This should be a fun assignment.

Successful programming languages tend to evolve over time, especially in the context of their community. Some languages use standards committees to represent this community; others use more informal processes. Either way, any potential evolution is frequently encapsulated into a change proposal, which the community can evaluate, discuss, and further enhance. Frequently the change proposal is rejected outright, but sometimes it may live on in some other way, or even influence another language.

Choosing a topic

In this assignment you will pick a language you know well, or at least have some significant exposure to the domain it is often used in. However, to constrain the language choices for this assignment, your chosen language must be in the Tiobe top 50 languages list. However some languages in that list will not be good choices. Consult the resources section, updating as necessary.

Once picked, find a reasonably substantial change proposal for that language and choose that as your paper topic. This change proposal may be currently in a proposed state, or it may now be rejected, accepted, or implemented. You may find it most straightforward to write about accepted proposals that are now implemented. However, rejected proposals may be of especial interest if the language later returned to that proposed idea, but in some other way.

Change proposals will be in the form of a PEP, SIP, JEP, JSR, or similar document. You may want consider subsequent proposals that cite the proposal as part of the development of your theme. See below for resources.

And then what? You need to carefully consider the change proposal in terms of documenting its proposed impact to the language and its use.

Writing your paper

Divide into teams of three, using the partner selection tool in Piazza as necessary. You may choose your lab partners for any previous lab - this is an independent selection.

Your paper on this change proposal should exhibit the following characteristics:

- Apply the terminology and concepts we have used throughout the course. If language feature Baz supports higher-order functions, and takes advantage of nested scopes and currying, state so in such terms and concepts. If

feature Bar provides syntax that makes certain common tasks easier or less error prone in the language, perhaps describe the syntax in the context of allowed grammar productions.

- Provide supporting code examples, diagrams, and other evidence. Such examples can be original to you or can come from the change proposal and any corresponding discussion.
- Cite supporting evidence using primarily primary sources from the community of the language being investigated.
- As usual, academic standards of plagiarism and attribution do apply.

When citing evidence, you should look for authority over form. For this paper, blog posts and their comments, emails in mailing lists, and Q&A in StackOverflow-type sites are valid primary sources, even if informal, along with articles and books. What matters here is the content of the discussion and where it went. Consider carefully the author of the source, and its centrality, in the conversation. In addition, do filter out noise and irrelevance. Careful selection and presentation are important parts of what we will evaluate.

Papers must be written in Markdown, specifically using the GitHub dialect. However, you are allowed to embed Latex for mathematical notation, as you see fit, using the following convention from Pandoc: `$ <latex goes here> $`. (GitHub used to render such notation with MathJax, and it's a major missing feature for us at this time.)

Papers must be between 1500 and 2000 words, no more, no less, excluding code fragments, long quotations (introduced via Markdown's `>`), and other metadata, such as images or citations links.

Use Markdown fences to demarcate code examples:

````
```python
def foo():
   return 42
```
````

Under no circumstances should you embed code samples in your paper or your presentation as an image! (This admonition is based on previous experience. But seriously it's so much easier to do it the right way. . .)

Use a backtick to mark a short fragment (under a line of code), such as `` `foo ` ``.

Introduce long quotes with `>`. Use endnote style links: use `[text][optional-id-if-different]` in the text, followed by an endnote with corresponding link at the end.

More on using Markdown in your paper or presentation can be found at this useful Markdown cheatsheet.

Presenting your paper

Now take the work on your group paper and distill it down to something to present to the class in a five minute slot:

- 20 unique slides, spaced *exactly* 15 seconds apart, for a total of 5 minutes.
- This is a class on programming languages, so show and talk about code! You want to find a good balance between describing a concept abstractly and demonstrating specific code usage in the language you're writing about in your paper.
- Slides must be written in Markdown, however, they can embed other media or use embedded Latex, much like the lecture notes do.
- Presentation is a PDF file generated via Beamer and pandoc; see the below section.
- Everyone on the team must be part of the presentation.

This style we are requiring for this presentation is actually closest to the Ignite format, vs a typical conference lightning talk. I recommend this blog post as a good start: it walks you through the process of going from a storyboard to a finished product. You may also find that a short presentation can sometimes require more work than a longer presentation: that's the essence of getting your content distilled down.

Presentation schedule

After the topic deadline, the instructors will generate a presentation schedule using the `shuffle` function from the `random` module in Python, such that groups will present in class on either December 9 or December 11. If you cannot make your time slot, at prior notice to your TA, you can instead submit the link of a video posted on YouTube of your presentation (this of course must also be exactly 5 minutes and conform to the lightning talk guidelines). Deadline for any videos posted on YouTube is **December 11** at 7pm.

Generating your presentation

Your presentation must be convertable to PDF by using the pandoc tool. You will use the same pipeline as is used for generating the lecture notes. Running

```
$ ./generate talk.md
```

will produce `talk.pdf`.

You can also generate a PDF for your paper if you want:

```
$ ./generate --format=paper paper.md
```

Installing pandoc on Ubuntu

It's simple on Ubuntu, as might be expected:

```
$ sudo apt-get install pandoc
$ sudo apt-get install latex-beamer
```

Installing pandoc on OS X

For OS X, download and install the following packages:

1. pandoc DMG at https://code.google.com/p/pandoc/downloads/
2. BasicTex at http://www.tug.org/mactex/morepackages.html
3. MacTeX-Additions also at http://www.tug.org/mactex/morepackages.html

Then do the following:

```
$ sudo tlmgr install pgf
$ sudo tlmgr install pgf-umlsd
$ sudo tlmgr install xstring
$ sudo tlmgr install smartdiagram
```

Help for generate

```
$ ./generate  --help
usage: generate [-h] [--format FORMAT] [--incremental] source

Simple driver for pandoc. Or use pandoc directly for more options.

positional arguments:
  source             Source markdown file

optional arguments:
  -h, --help         show this help message and exit
  --format FORMAT    Output format
  --incremental, -i  Incremental display of lists
```

Paper grading rubric

Your paper will be evaluated with respect to the expected characteristics. In particular, do not use personal opinion, unless it can be supported by evidence. Don't use fuzzy language. This means to not write sentences like the following: "I think feature Baz in language Foo is amazing because it's so helpful and consequently my code will never have errors again!"

Lightning talk grading rubric

- 80% content of the talk itself
- 20% of your constructive peer assessments using a Google form for the other talks given. You can either submit your entries while watching the presentation during class, or take notes and submit this later.

Lastly:

- Be creative.
- Remember to have fun!

Important dates and requirement

- Initial topic deadline on Friday November 14 at 7 pm. You must consult with your TA in advance for their approval (recitation, office hours, email). We want substantive and interesting topics. You are free to change this topic after this deadline. You will also be asked about your topic during the lab 4 interview grading slots.
- Talks on Tuesday December 9 or Thursday December 11 during normal classtime.
- Deadline for submitting comments about talks: Thursday December 11 at 7 pm.
- Final paper due on Thursday December 11 at 7 pm.

No credit will be given for papers/talks written/presented by groups less than 3 people. Only as necessary will we allow 4 person groups, but only with TA permission.

Resources

Here is a sample resource in where you should start in your investigation:

- Python uses PEPs (Python Enhancement Proposals) for language proposals. Two mailing lists are typically used, python-ideas and python-dev to discuss PEPs and their implementation. In addition, you may find the "What's new in Python x.y" useful in understanding the scope of changes; you may want to look at series of these "What's new" summations.

Other languages will have similar places to look. Since you have a choice of 49 other languages, we will not attempt to enumerate them.

Extra credit opportunities

We welcome your contributions to better defining this assignment. Extra credit, in proportion to the work submitted, will be awarded for discussion under the **project tab** in Piazza. Here are some potential ideas; you might have others:

- Identifying resources for investigating a specific language.
- Finding typos and other clarity issues in this assignment. How to identify a clarity issue? Take a look at the Piazza questions that are likely to be made, or consider the conversation you have with your classmates and instructors about this assignment.
- Suggest grading criteria. What would make for a strong paper vs a weak paper?
- Tools that can assist in working on this paper or presentation.