Cultural differences in Programming Language Communities

Week 6

Timetable

09.02. - Finish Literature Review (with rough draft of text)

16.02. - Finish Encoding

23.02. - Finish Data Analysis

??? - Present in AG SE?

07.03. - Hand in a beautiful thesis (Deadline: 08.03.)

??? - Defence

Theme 1

Answer: I personally like the solution using the parser module, which is the second answer to this question and is beautiful, as you don't have to construct any string literals to get it working. **But**, one downside is that it is **90% slower** than the accepted answer with strptime. [...]

Answer: Either do: local $@; eval { … }

… to prevent changes to $@ from affecting global scope, or use Try::Tiny. [...]

Syntactically, there are situations where I prefer one or the other.

Comment: @KarlKnechtel I miss having "some way of referencing symbols in one file from another" that wasn't designed by a monkey on crack. I also miss being able to choose what symbols to expose to other files vs Python's "publish everything".

Question: [...] What are the differences between these methods? Which method should be preferred?

Comment: .sample is such an non obvious method name. I loved choice much better. Change it back Ruby developers!

One way vs. Many

Answer: I personally like the solution using the parser module, which is the second answer to this question and is beautiful, as you don't have to construct any string literals to get it working. **But**, one downside is that it is **90% slower** than the accepted answer with strptime. [...]	Python	Approval of multiple options
Answer: Either do: local $@; eval { … } … to prevent changes to $@ from affecting global scope, or use Try::Tiny. [...] Syntactically, there are situations where I prefer one or the other.	Perl	Approval of multiple Options
Comment: @KarlKnechtel I miss having "some way of referencing symbols in one file from another" that wasn't designed by a monkey on crack. I also miss being able to choose what symbols to expose to other files vs Python's "publish everything".	Python	Disapproval of recommended solution
Question: [...] What are the differences between these methods? Which method should be preferred?	Python	Asking for recommended solution
Comment: .sample is such an non obvious method name. I loved choice much better. Change it back Ruby developers!	Ruby	Disapproval of recommended solution

Theme 2

Answer: [...] That by itself won't magically bring your Python script in line with your Perl script, but repeatedly calling re in a loop without compiling first is bad practice in Python.

Comment: good technique with the mention that it pays off only when multiple lookups are made. +1

Comment: Just a comment. Ruby's a bit of a pain about naming conventions but not really obvious about them until you trip over them. The names of the enums must be all caps and the first letter of the module name must be capitalized for ruby to know that the module is a module of constants.

Answer: [...] In the end, what worked for me was not pythonic, I guess, but:

I used a if __main__ on top of the module I wanted to debug, that is run from a different than usual path.

Comment: @emk, thanks for your comment, it really sums up the dynamic-lang crowd's thought on the thing. While I do think it's wrong, that's how it works :)

What should idiomatic Code look like?

Answer: [...] That by itself won't magically bring your Python script in line with your Perl script, but repeatedly calling re in a loop without compiling first is bad practice in Python.	**Python** Calling something bad practice or style
Comment: good technique with the mention that it pays off only when multiple lookups are made. +1	**Perl** Calling something good practice or style
Comment: Just a comment. Ruby's a bit of a pain about naming conventions but not really obvious about them until you trip over them. The names of the enums must be all caps and the first letter of the module name must be capitalized for ruby to know that the module is a module of constants.	**Ruby** Implicit rules or conventions
Answer: [...] In the end, what worked for me was not pythonic, I guess, but: I used a if __main__ on top of the module I wanted to debug, that is run from a different than usual path.	**Python** No subcode
Comment: @emk, thanks for your comment, it really sums up the dynamic-lang crowd's thought on the thing. While I do think it's wrong, that's how it works :)	**Ruby** Disapproval of convention or implicit rules

Miscellaneous

Comment: [...] I can understand why you make your arguments, but it requires that a programmer never forget to check for errors. We know they should, but what we should do in theory and what we do in practice when under the gun of a tight deadline aren't the same thing. Thus, we could forget to check for that one critical error and have our code continue on, blithely unaware that it's corrupting its data everywhere.

Answer: [...] It also requires exactly two arguments to be passed in (aside from the self.) No more, no less. That's an interesting constraint that future users may not appreciate.

To be direct - it violates Liskov substitutability.

Comment: why not extend class Array ? [].total_random would be way cooler. comeon its ruby. its objective!

Answer: [...] There's really no need to write your own __str__ or __repr__. The built-in ones are very nice, and your cooperative inheritance ensures that you use them. [...]

Miscellaneous

Comment: [...] I can understand why you make your arguments, but it requires that a programmer never forget to check for errors. We know they should, but what we should do in theory and what we do in practice when under the gun of a tight deadline aren't the same thing. Thus, we could forget to check for that one critical error and have our code continue on, blithely unaware that it's corrupting its data everywhere.	Perl Pragmatism vs. Purity -> Suggesting a pragmatic approach
Answer: [...] It also requires exactly two arguments to be passed in (aside from the self.) No more, no less. That's an interesting constraint that future users may not appreciate. To be direct - it violates Liskov substitutability.	Python Pragmatism vs. Purity -> Concern for future Coders Referencing CS theory
Comment: why not extend class Array ? [].total_random would be way cooler. comeon its ruby. its objective!	Ruby Talk of general coding principles -> The OOP way
Answer: [...] There's really no need to write your own __str__ or __repr__. The built-in ones are very nice, and your cooperative inheritance ensures that you use them. [...]	Python Talk of general coding principles -> Don't reinvent the wheel

Overview Coding System - Main Themes

- **One Way vs. many:** Do developers think there should be (mainly) one correct approach to a solution, or can many solutions be good and correct.

  "There's a preferred solution", "(Dis)approval of multiple options", "Mentioning of Aliases", "Answer gives additional or multiple options"

- **Implicit vs. Explicit:** Python famously has the mantra of explicitness, Ruby says 'convention over configuration'. How do developers feel and think about this?

  "Implicit rules or conventions", "Unwritten Rules of Good Coding", and there should be more...

- **Idiomatic Code:** What does the community consider to be idiomatic?

  "Spirit of the language", Calling something good/bad practice or style, "preferred style of solution"

Overview Coding System - Additional

- **Pragmatism vs. Purity:** Do developers hold abstract principles highly, is there a strong relation to CS Theory, or is quick and dirty acceptable

  “Concern for future Coders”, “Referencing CS Theory”, “Lazy approach is good”, “Messy (but it works)”

- **Legacy vs. modern practices:** How does the community deal with issues of past and future concerns

  “Inquiring about modern approaches”, “Don’t do the deprecated”, “Don’t mind if I do the deprecated”, “Mentioning of future options or concerns”, “Importance of Backwards Compatibility”

Other categories: “Talk of abstract coding principles”, “Comparison to other programming languages”, “Usage of Sources”, “Conversational Tone”, “What is Efficiency”, “What is Beauty”, “Ideas on packages and when to avoid them”