PySonar2 与 Sourcegraph 集成完毕

来到 Sourcegraph 两个星期了,我可以说这里的每一天都是激动人心的,这是一个有真正创造活力的 startup。我们的发展速度相当之快,每一天都出现新的点子,或者发现以前做法的一些大幅度简化。不得不承认 Quinn 和Beyang 是比我有魄力的人。我虽然做出了 PySonar,却让它的代码束之高阁多年之久,没有发挥出应有的作用。而Quinn 和 Beyang 看准了这种东西的价值,坚持不懈地做出了 <u>Sourcegraph.com</u> 这个网站,才使 PySonar 可以发挥出这么强劲的效果,用以搜索全世界的 Python 代码,为广大程序员造福。当然,我们的目标不只限于Python。Sourcegraph 目前支持 Go, JavaScript, Python 和 Ruby。其中 Ruby 的支持还处于初步阶段,需要改善,更多其它的语言正在开发中。

经过两个星期的勤奋却又不知疲倦的工作,PySonar2 的代码得到了巨大的改善。现在我可以很自信的说,它包含了世界上最强大的静态分析技术。今天 PySonar2 已经正式与 Sourcegraph.com 集成完毕。现在只要登录 Sourcegraph 主网站就可以看到开源 Python 代码的 PySonar2 分析结果。

PySonar2 的类型推导系统能够不依赖类型标记却精确地分析出 Python 函数的参数类型。比如下图所示的 Flask 框架的最常用的五个函数的参数,都是用通常的方法很难确定类型的,PySonar2 却能得知它们的正确用法。

Top Functions		Code Browser	README
method	Flask.route(self, rule, **options) (Flask, str) -> () -> str -> () -> str A decorator that is used to register a view function for a given URL rule.		
func	<pre>templating.render_template(template_name_or_list, **context) str -> ? I [str] -> ? Renders a template from the template folder with the given context.</pre>		
func	helpers.url_for(endpoint, **values) str -> ? Generates a URL to the given endpoint with the method provided.		
func	helpers.flash(message, category) (str, str) -> None I (Markup, str) -> None Flashes a message to the next request.		
method	Blueprint.route(self, rule, **options) (Blueprint, str) -> ? -> ? Like :meth:`Flask.route` but for a blueprint.		

最有意思的是那个 render template。PySonar2 为它推导出来的类型是一个 intersection type:

```
templating.render_template(template_name_or_list, **context)
str -> ?
| [str] -> ?
```

这是说,第一个参数 template_name_or_list 的类型或者是 str 或者是 [str] (含有 str 的 list)。如果你给它 str 它就会输出?(PySonar2 不知道它会输出什么),而如果你给它 [str],它输出?.

如果你注意一下这个参数的英文含义 "template name or list",就觉得仿佛 PySonar2 能读懂英文一样。然而 PySonar2 其实不会英文,它只会 Python,它通过代码之间的调用关系和异常强大的类型推导,找到了这个参数的 类型。

Sourcegraph 的一些使用诀窍

Sourcegraph 有一些不为人知的巧妙设计,但是由于 Quinn 和 Beyang 太谦虚而且太忙了,所以都没来的及宣传。我现在偷闲在这里透露两招小窍门。

启动分析你需要的 GitHub 代码库

目前这个功能只限于 GitHub。如果在 Sourcegraph 网站上面没有找到你需要的 GitHub 代码库,这不等于你需要等我们来启动分析。你可以自己动手!

方法很简单:把你的 GitHub 地址去掉 http://之后放到 http://sourcegraph.com/ 后面,然后 Sourcegraph 就会显示一个等待页面,同时自动开始分析这个 repo。



举个例子,如果你想分析 http://github.com/myname/myrepo 的代码,就在浏览器输入地址:

http://sourcegraph.com/github.com/myname/myrepo

如果 Sourcegraph 还没有分析过这个 repo 它就会把它加入到工作队列里,然后你可以做其他的事情或者浏览其他的代码。分析完毕之后浏览器就会自动跳转到你所需要的代码库。一般大小的代码库几分钟到半个小时就会处理完毕。

在你的 GitHub README 里面使用 Sourcegraph 徽章

你也许发现有些人在自己的 GitHub 里有 Sourcegraph 徽章,这样一来别人就能得知你的代码库的一些统计信息。 比如我的 psydiff 的 README 里面有这样一个:

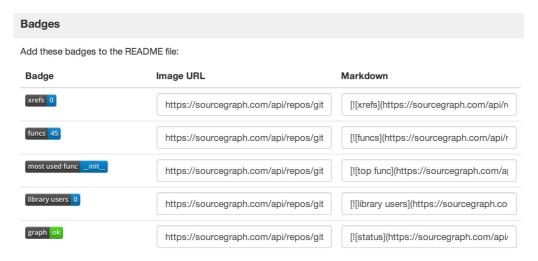


它表示 psydiff 的代码被看过的次数。你也可以使用其他的一些徽章,比如最常用的函数,交叉引用数,用户数,等等:



要得到这些徽章很简单,只要在你的 repo 的 Sourcegraph 主页里点击如图所示的扳手状小图标,然后把 "Image URL" 拷贝到你的网页里就行:





Sourcegraph 的功能虽然非常强劲,但是很多设计的工作还处于起步阶段。如果你有什么建议或者发现问题,请联系我们:hi@sourcegraph.com.