迟来的 Agda 环境搭建

2018, Jun 13 by Tesla Ice Zhang

前面讲了辣么多 Agda 的语言特性、写证明的思路等，但是都没有介绍 Agda 的具体编程方法，所以我先补上这个空缺。

首先，我们要搭建 Agda 开发环境。

流程是：

1. 在 apt 里安装合适版本的 ghc
2. 在 apt 里安装合适版本的 cabal/stack (本文使用 cabal, 因为我们不需要用 Haskell 写项目)
3. 在 apt 里安装合适版本的 Emacs
4. 在 cabal 里安装合适版本的 alex, happy, cpphs
5. 在 cabal 里安装最新版本的 Agda
6. 通过 git 安装最新版本的 Agda 标准库
7. 学习 Agda 编程方法

如果你已经熟悉 cabal/stack 和 ghc 的安装以及 Haskell 包的安装，那么你可以直接跳到 Emacs 的使用那一步。

安装 Agda

我写了一个用于初始化 Ubuntu 系统的脚本，用来安装我需要的所有程序和进行所有配置：

其中，我们需要执行的代码是：

```
## prerequisites
$ sudo apt install cabal ghc git emacs
$ export INSTALLATION_PATH=~/SDK # replace with your ow

## installing agda
$ cabal update
$ cabal install alex happy cpphs
$ cabal install --allow-newer Agda
$ agda-mode setup
$ agda-mode compile
$ mkdir -p $INSTALLATION_PATH
$ git clone https://github.com/agda/agda-stdlib.git $IN
$ rm $INSTALLATION_PATH/agda-stdlib/src/index.agda
$ mkdir ~/.agda
$ echo "$INSTALLATION_PATH/agda-stdlib/standard-library
$ echo "standard-library" >> ~/.agda/defaults
```

请先自行修改 `INSTALLATION_PATH` 为你希望的 Agda 标准库的安装目录，以及把 cabal 和 ghc 改为你需要的版本。 推荐的 cabal 版本是 2.2，ghc 版本是 8.2.2。

安装完成后，可以这样测试：

```
$ agda --version
```

如果输出了版本信息，说明安装正常。

配置 Emacs

如果你真的非常不能接受 Emacs，你可以去官网找 Atom 环境搭建的教程。 由于我本身就习惯使用 Emacs 编程了，所以就只讲 Emacs 了。
如果你是照着上一步来的，你执行了所有刚才给的 shell 脚本，那么就不需要这一步。 否则，你需要执行：

```
$ agda-mode setup
$ agda-mode compile
```

然后你可以使用 Emacs 打开一个 `.agda` 文件（更多关于 Emacs 的使用技巧，请参考我的这篇文章）：

```
emacs Hello.agda
```

打开之后先输出以下代码（`module` 名必须和文件名匹配）：

```
module Hello where
```

按下 Ctrl + C Ctrl + L，可以看到高亮出现了， 屏幕下方还出现了一个滴点儿大的 buffer，里面什么都没有。
如果你不能理解这个很长很长的快捷键，你可以用我按这个快捷键的方法去按：

1. 按住 Ctrl

2. 按下 C
3. 松开 C
4. 按下 L
5. 松开 L
6. 松开 Ctrl

高亮出现代表 Parsing 和基本的检查通过了，每次刷新高亮都需要按这个快捷键，刚输入的字符的颜色是错误的。
如果你的代码有词法上的错误，那么出错的地方会有红色高亮，其他地方是黑色的。
如果你的代码有在 exhaustiveness, termination 上有问题，那么出错的地方会有黄色背景，其他地方是正常高亮的。
这两种情况下，下面那个滴点儿大的 buffer 会有错误信息。

写点代码吧

我们可以试着写点代码。比如，定义一个 GADT:

```
data List (A : Set) : Set where
  [] : List A
  _::_ : A -> List A -> List A
```

按 Ctrl + C Ctrl + L 出现高亮。 然后我们编写一个函数，并不写实现：

```
emptyList : {A : Set} -> List A
emptyList = ?
```

对，你没有看错，我们需要在代码里写下这个问号，表示我们暂时不确定这个地方可以写什么。 Agda 强大的类型系统拥有辅助程序员填写问号内的内容的功能，有时还可以直接自己填写实现。

按 Ctrl + C Ctrl + L 再次出现高亮，你会发现代码变成了这样：

```
emptyList : {A : Set} -> List A
emptyList = { }0
```

这个绿色背景的 `{ }0` 就是 Agda 的『洞』，表示一个还没想好怎么写的表达式，在下面的滴点儿大的 buffer 里注明了它的类型。 把光标放在那个『洞』里面，按 Ctrl + C Ctrl + A ，你会发现 Agda 给这个洞填入了一个值。

```
emptyList : {A : Set} -> List A
emptyList = []
```

Idris, Coq 都具有这样的功能。

我说完啦。

为什么要写这篇文章

最近被 @16 拉着写的 Agda 代码比你们想象的要多， 又产生了写博客的欲望。

我的计划是，先使用 Literate Agda 重写之前的几篇博客，摆脱傻逼 LaTeX 代码，然后继续更新这个系列。

更新会发布到 zju lambda 的网站（如果龙神给这个网站续命了的话），我的博客，和知乎。

用 Literate Agda 重写的博客不会重新发布到知乎。

Tweet this

Top

创建一个 issue 以申请评论

Create an issue to apply for commentary

协议/License