0XHEXADECIMAL

HEXAGON 编程指南

内部文档

概览

Hexagon 是一门全新的编程语言。它具有全新的完美类库分离语法、完全函数编程支持、强面向集合性等诸多令人喜爱的特性以及一些经过改善的语法。一睹为快吧。

0.1 Hello, world!

将以下代码输入一个文件中，把它重命名为 HelloWorld.Hexagon。

清单 1: 你好，世界

```
Hexagon::Result() Main = (){
        var c = import Hexagon::System::Console;
        c << "Hello, World!" << Hexagon::System::SpecChar::EndLine;
        c.Command << "PAUSE";
        return Hexagon::Result.Fine;
};
```

打开控制台，运行

```
$ hgc HelloWorld.Hexagon
<编译提示>
$ hgx HelloWorld.Hexec
```

产生输出：

```
Hello, World!
```

目录

插图目录

表格目录

清单目录

第一部分

入门篇

第 1 章

学校成绩数据库

在本章中，我们将为您展示一个十分基本的程序供您参阅和研究。这个程序将会具有以下的功能：

- 运行时按行读取输入
- 录入学生的成绩
- 查询学生的成绩

接下来，让我们一起来实现它吧。

相关代码可在 < 待填充 > 下载。

1.1 程序起点

我们的程序的起点在一个叫做 *Main*的函数里。

清单 1.1: main 函数的标准形式

```
Hexagon::Result() Main;                                    //声明
Main = (){
        //要执行的语句
        return Hexagon::Result.Fine;     //返回程序的运行结果
};
```

我们可以把一条至多条语句放到要执行的语句的位置。如果您是第一次接触这种风格，或是第一次接触编程语言，您可以暂时不理解它们的具体含义。您只需将这段代码反复利用即可。要执行的语句就是程序的起点。

1.2 命令行输入/输出

您可以用import Hexagon::Console 来导入所谓控制台输入/输出流。我们将这样使用它：

清单 1.2: 输入/输出流

```
import Hexagon::Console;
Console << "这是我们提供的输出流。";
Console << "请输入一个整数："
Hexagon::Interger i;
Console >> i;   //这是输入的方式
```

有关于 *import* 命令和输入/输出流，我们将在以后具体描述，目前您只需了解如何基本地使用它们即可满足需要。

1.3 语句

我们在执行一个动作后，用分号“;”来结束表述，放弃现有的运算结果。

1.3.1 空语句

如果我们直接使用一个分号，而前面没有任何表达式，这时这个语句成为了一个空语句。语法上来讲，这种做法是允许的；但是为了保持我们的程序尽可能的简洁，而且容易被别人读懂，请直接删除空语句，因为空语句和什么都没有其实是等效的 [*]。

[*] Hexagon 中没有任何获取上下文语句的情况，所有的控制流都是直接传入函数对象，这一点与 C/C++ 等语言不大相同。

1.4 基本运算

我们提供几个十分基础的算术类型，它们代表一些数学意义上的实数、整数等基本情况如**??**所示。

这些类型提供四则运算、乘开方、求对数等基本的数字功能，我们将在各种程序中直接使用它们。

[†] 有关于其他进制表示我们将在以后讨论，本表一律采用十进制。

[‡] 它只表示一个近似的示数。

[§] 此处表示的是，本类型可选一个循环节部分，写作“整数部分. 小数部分. 循环节”。

表格 1.1: 基础算术类型一览

类型名	中文名	说明	示例	一般形式
Interger	整数型	表示一个任意尺寸的整数。	10, 35, 81263845618	(1 至 9) 后接 ((0 至 9) 重复 0 或多次)†
Number	实数型‡	表示一个任意尺寸的自然数。	1.0, 3.14, 1.9773.827§	(整数型) 后接 ((.(整数型)) 重复 0 至 2 次)
Complex	复数型	表示一个由两个实数型构成的复数。	i, 3.5i, 2i+1	(实数型 i) 后接 (可选的 (+ 实数型)

第 2 章

表达式与语句

在本章里我们将研究两个极为重要的概念：能够描述一段运算的表达式和一个动作的语句。事实上，语句是表达式的一个超集。

2.1 变量

2.2 运算

表达一定的数值上的计算一般被叫做运算。例如

```
1+1
5+3
(25-3)/2+7
```

运算常常是字面量和运算符所进行的操作。

第 3 章

函数

第 4 章

面向对象

第二部分

进阶篇

第 5 章

泛型

有的时候我们想要制作一种基于一个类的类，例如，一个整数数组。

第 6 章

面向集合

第 7 章

数据结构

第三部分

提高篇

第 8 章

算法

第 9 章

自订模块

附录

附录 A

索引