基础概念篇

1. Go 语言在多个工作区中查找依赖包的时候是以怎样的顺序进行的？

答：你设置的环境变量GOPATH的值决定了这个顺序。如果你在GOPATH中设置了多个工作区，那么这种查找会以从左到右的顺序在这些工作区中进行。你可以通过试验来确定这个问题的答案。例如：先在一个源码文件中导入一个在你的机器上并不存在的代码包，然后编译这个代码文件。最后，将输出的编译错误信息与GOPATH的值进行对比。

2. 如果在多个工作区中都存在导入路径相同的代码包会产生冲突吗？

答：不会产生冲突。因为代码包的查找是按照已给定的顺序逐一地在多个工作区中进行的。3. 默认情况下，我们可以让命令源码文件接受哪些类型的参数值？

答：这个问题通过查看flag代码包的文档就可以回答了。概括来讲，有布尔类型、整数类型、浮点数类型、字符串类型，以及time.Duration类型。

4. 我们可以把自定义的数据类型作为参数值的类型吗？如果可以，怎样做？

答：狭义上讲是不可以的，但是广义上讲是可以的。这需要一些定制化的工作，并且被给定的参数值只能是序列化的。具体可参见flag代码包文档中的例子。

5. 如果你需要导入两个代码包，而这两个代码包的导入路径的最后一级是相同的，比如：dep/lib/flag和flag，那么会产生冲突吗？

答：这会产生冲突。因为代表两个代码包的标识符重复了，都是flag。

6. 如果会产生冲突，那么怎样解决这种冲突？有几种方式？

答：接上一个问题。很简单，导入代码包的时候给它起一个别名就可以了，比如： import libflag "dep/lib/flag"。或者，以本地化的方式导入代码包，如：import . "dep/lib/flag"。7. 如果与当前的变量重名的是外层代码块中的变量，那么意味着什么？

答：这意味着这两个变量成为了“可重名变量”。在内层的变量所处的那个代码块以及更深层次的代码块中，这个变量会“屏蔽”掉外层代码块中的那个变量。

8. 如果通过import . XXX这种方式导入的代码包中的变量与当前代码包中的变量重名了，那么 Go 语言是会把它们当做“可重名变量”看待还是会报错呢？

答：这两个变量会成为“可重名变量”。虽然这两个变量在这种情况下的作用域都是当前代码包的当前文件，但是它们所处的代码块是不同的。当前文件中的变量处在该文件所代表的代码块中，而被导入的代码包中的变量却处在声明它的那个文件所代表的代码块中。当然，我们也可以说被导入的代码包所代表的代码块包含了这个变量。在当前文件中，本地的变量会“屏蔽”掉被导入的变量。9. 除了《程序实体的那些事儿 3》一文中提及的那些，你还认为类型转换规则中有哪些值得注意的地方？答：简单来说，我们在进行类型转换的时候需要注意各种符号的优先级。具体可参见 Go 语言规范中的转换部分。10. 你能具体说说别名类型在代码重构过程中可以起到的哪些作用吗？答：简单来说，我们可以通过别名类型实现外界无感知的代码重构。具体可参见 Go 语言官方的文档 Proposal: Type Aliases。数据类型和语句篇11. 如果有多个切片指向了同一个底层数组，那么你认为应该注意些什么？答：我们需要特别注意的是，当操作其中一个切片的时候是否会影响到其他指向同一个底层数组的切片。如果是，那么问一下自己，这是你想要的结果吗？无论如何，通过这种方式来组织或共享数据是不正确的。你需要做的是，要么彻底切断这些切片的底层联系，要么立即为所有的相关操作加锁。12. 怎样沿用“扩容”的思想对切片进行“缩容”？答：关于切片的“缩容”，可参看官方的相关 wiki。不过，如果你需要频繁的“缩容”，那么就可能需要考虑其他的数据结构了，比如：container/list代码包中的List。13. container/ring包中的循环链表的适用场景都有哪些？答：比如：可重用的资源（缓存等）的存储，或者需要灵活组织的资源池，等等。14. container/heap包中的堆的适用场景又有哪些呢？答：它最重要的用途就是构建优先级队列，并且这里的“优先级”可以很灵活。所以，想象空间很大。15. 字典类型的值是并发安全的吗？如果不是，那么在我们只在字典上添加或删除键 - 元素对的情况下，依然不安全吗？答：字典类型的值不是并发安全的，即使我们只是增减其中的键值对也是如此。其根本原因是，字典值内部有时候会根据需要进行存储方面的调整。16. 通道的长度代表着什么？它在什么时候会通道的容量相同？通道的长度代表它当前包含的元素值的个数。当通道已满时，其长度会与容量相同。17. 元素值在经过通道传递时会被复制，那么这个复制是浅表复制还是深层复制呢？答：浅表复制。实际上，在 Go 语言中并不存在深层次的复制，除非我们自己来做。18. 如果在select语句中发现某个通道已