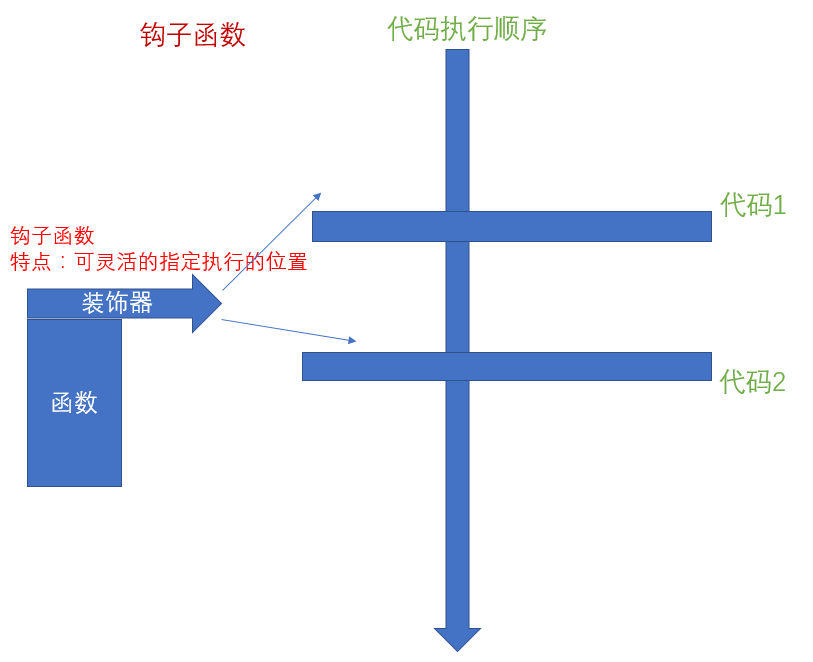
**Flask\_钩子函数概念\_常见的钩子函数：**

**1.钩子函数概念：**

在Flask中钩子函数是使用特定的装饰器装饰的函数。为什么叫做钩子函数呢，是因为钩子函数可以在正常执行的代码中，插入一段自己想要执行的代码。那么这种函数就叫做钩子函数。



**2.常见的钩子函数：**

**before\_first\_request：**处理项目的第一次请求之前执行。

@app.before\_first\_request
def first\_request():
print 'first time request'

**before\_request：**在每次请求之前执行。通常可以用这个装饰器来给视图函数增加一些变量。请求已经到达了Flask，但是还没有进入到具体的视图函数之前调用。一般这个就是在视图函数之前，我们可以把一些后面需要用到的数据先处理好，方便视图函数使用。

@app.before\_request
def before\_request():
if not hasattr(g,'glo1'):
setattr(g,'glo1','想要设置的')

**teardown\_appcontext：**不管是否有异常，注册的函数都会在每次请求之后执行。

@app.teardown\_appcontext
def teardown(exc=None):
if exc is None:
db.session.commit()
else:
db.session.rollback()
db.session.remove()

**template\_filter：**在使用Jinja2模板的时候自定义过滤器。

@app.template\_filter("upper")
def upper\_filter(s):
return s.upper()

**context\_processor：**上下文处理器。使用这个钩子函数，必须返回一个字典。这个字典中的值在所有模版中都可以使用。这个钩子函数的函数是，如果一些在很多模版中都要用到的变量，那么就可以使用这个钩子函数来返回，而不用在每个视图函数中的`render\_template`中去写，这样可以让代码更加简洁和好维护。

@app.context\_processor
def context\_processor():

if hasattr(g,'user'):

return {"current\_user":g.user}

else:

return {}

**errorhandler：**errorhandler接收状态码，可以自定义返回这种状态码的响应的处理方法。在发生一些异常的时候，比如404错误，比如500错误,那么如果想要优雅的处理这些错误，就可以使用`errorhandler`来出来。

@app.errorhandler(404)
def page\_not\_found(error):
return 'This page does not exist',404