自动化测试框架（关键字驱动）

讲师：芳姐 | 时间：2018-11-27

什么是关键字驱动框架

关键字驱动框架是一种功能自动化测试框架，它也被称为表格驱动测试或者基于动作字的测试。关键字驱动的框架的基本工作是将测试用例分成四个不同的部分。
首先、是测试步骤（Test Step）

二、测试步骤中的对象（Test Object）

三、测试对象执行的动作(Action)

四、测试对象需要的数据（Test Data）

我们做关键字的驱动的思想，就是把编码从测试用例和测试步骤中分离出来，这样对于不会编码的人员更容易理解自动化，从而让手工测试人员也可以编写自动脚本。（这并不意味这不需要自动化测试人员，对于自动化框架的构建，自动化代码的更新，结构调整等都需要一个技术性的人员）对于测试小的项目的团队，可以有两个手工测试人员和一个自动化测试人员。

以上四个部分，都可以使用Excel表格进行维护:

Test Step：是一个小的测试步骤的描述或者测试对象的一个操作说明。

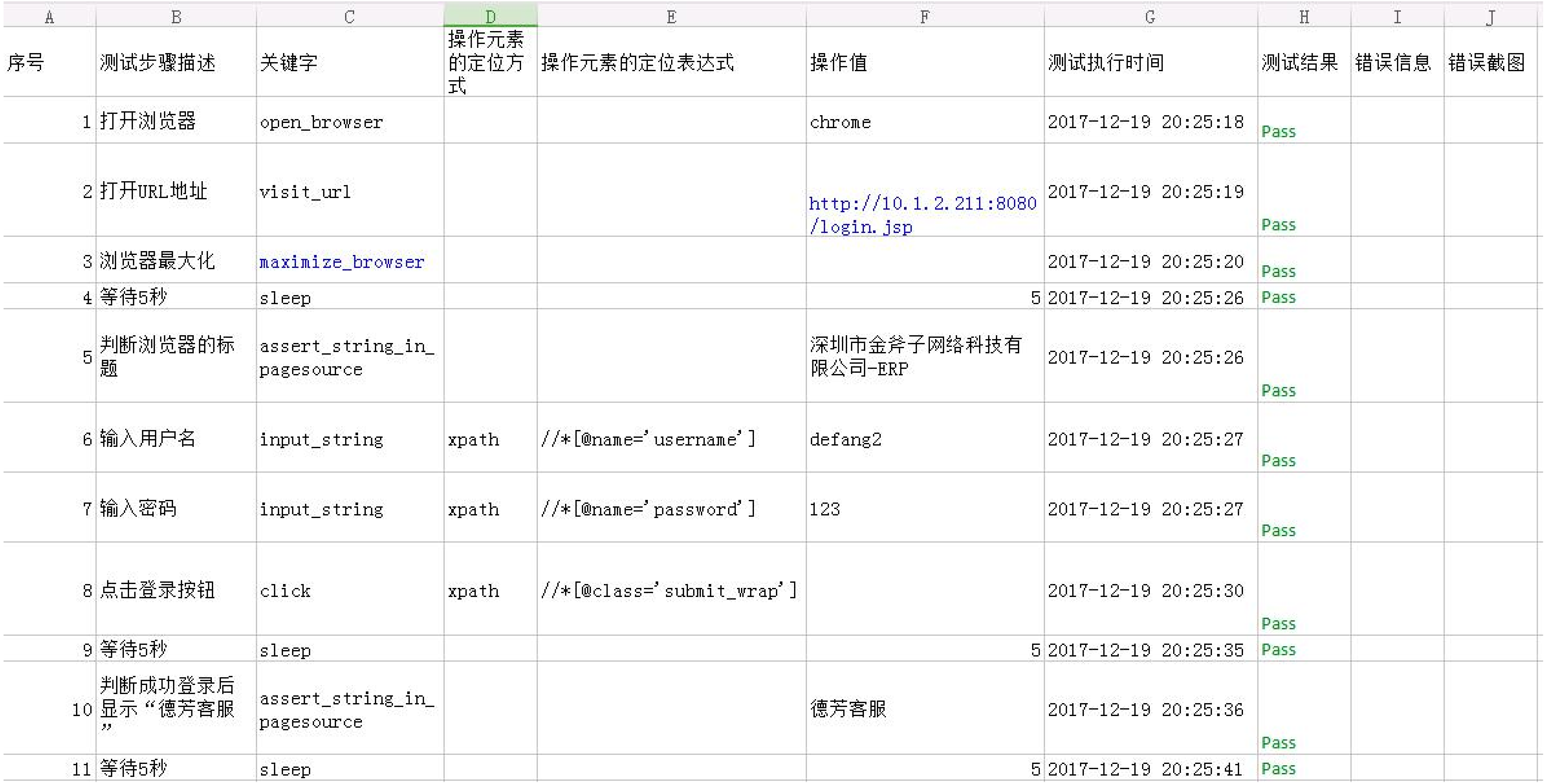
Test Object：是指页面对象或元素，就像用户名、密码，

Action：指页面操作的动作，打开浏览器，点击一个按钮，文本框输入一串文本等。

Test Data:是任何对象操作时所需要的值，就像用户名、密码进行输入时的输入内容

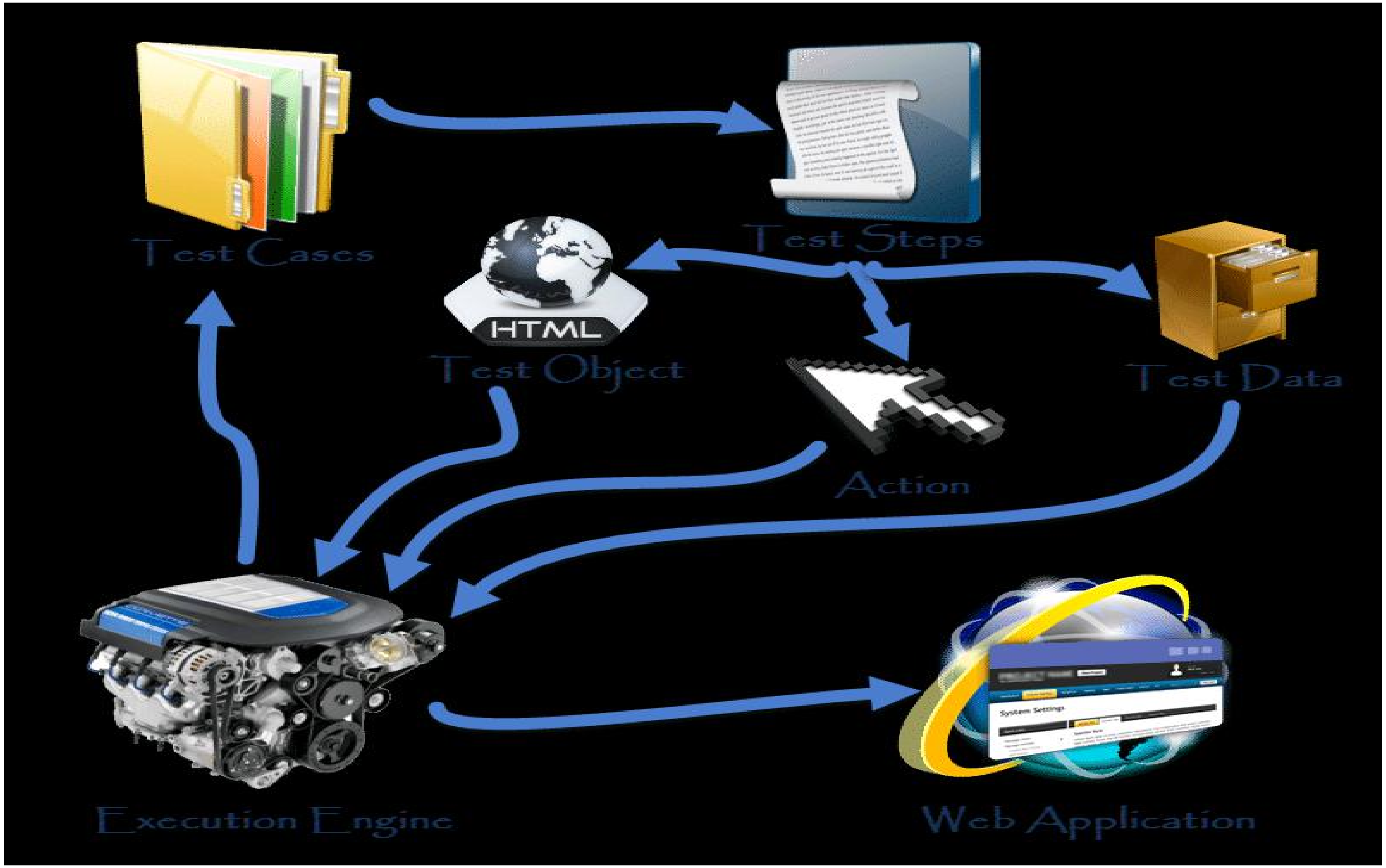
下面通过一个简单的登录功能例子来理解这个概念，想想看你的自动化流程需要做哪些事情：

1. 打开一个浏览器
2. 输入url跳转到网站首页
3. 点击“登录”链接，进入登录页面
4. 输入“用户名”
5. 输入“密码”
6. 点击“登录”按钮，进行登录
7. 点击“注销”按钮，退出登录
8. 关闭浏览器



A	B	C	D	E	F	G	H	I	J
序号	测试步骤描述	关键字	操作元素的定位方式	操作元素的定位表达式	操作值	测试执行时间	测试结果	错误信息	错误截图
1	打开浏览器	open_browser			chrome	2017-12-19 20:25:18	Pass		
2	打开URL地址	visit_url			http://10.1.2.211:8080/login.jsp	2017-12-19 20:25:19	Pass		
3	浏览器最大化	maximize_browser				2017-12-19 20:25:20	Pass		
4	等待5秒	sleep			5	2017-12-19 20:25:26	Pass		
5	判断浏览器的标题	assert_string_in_pagesource			深圳市金斧子网络科技有限公司-ERP	2017-12-19 20:25:26	Pass		
6	输入用户名	input_string	xpath	//*[@name='username']	defang2	2017-12-19 20:25:27	Pass		
7	输入密码	input_string	xpath	//*[@name='password']	123	2017-12-19 20:25:27	Pass		
8	点击登录按钮	click	xpath	//*[@class='submit_wrap']		2017-12-19 20:25:30	Pass		
9	等待5秒	sleep			5	2017-12-19 20:25:35	Pass		
10	判断成功登录后显示“德芳客服”	assert_string_in_pagesource			德芳客服	2017-12-19 20:25:36	Pass		
11	等待5秒	sleep			5	2017-12-19 20:25:41	Pass		

下面图片展示了关键字框架的通用工作流程：



一般的关键字驱动工作流程大概都是这个样子的。

Execution Engine starts the test and connect with the bundle of test cases and start executing one by one
（使用 Execution Engine 启动测试用例包，启动一个执行一个。）
Once Test Case is picked, linked test steps are followed sequentially
（一旦测试用例被选中，就会链接测试步骤顺序执行）
Test Steps are further connected with Page Objects, Actions & Test Data
（测试步骤会进一步链接页面对象，操作动作和测试数据）
Once Execution Engine gets all the required info to perform a test step, it connects with application and do the step.
（一旦Execution Engine获取到执行测试步骤所需的所有信息，他就会连接应用程序执行步骤）

优势

1. 不需要太多的技术：一旦框架建立，手工测试人员和非技术人员都可以很容易的编写自动化测试脚本。
2. 简单易懂：它存在Excel表格中，没有编码，测试脚本容易阅读和理解。关键字和操作行为这样的手工测试用例，使它变得更容易编写和维护。
3. 早期介入：可以在应用未提交测试之前，就可以建立关键字驱动测试用例对象库，从而减少后期工作。使用需求和其它相关文档进行收集信息，关键字数据表可以建立手工测试程序。
4. 组件的重用性：实施关键字驱动的模块化，进一步提高可重用性。
5. 代码的重用性：作为关键字驱动框架中，只有一个执行引擎，它是鼓励极端的代码的复用。

1、新建一个工程为KeyWordFrameWork的文件

- config包，主要用于实现框架中各种配置
- util包，主要用于实现测试过程中调用的工具类方法，例如读取配置文件，MapObject，页面元素的操作方法，解析excel文件
- testData目录，主要存放框架所需要的测试数据文件
- testScripts包，用于实现具有测试逻辑的测试脚本
- action目录：用于实现具体的页面动作，比如输入框输入数据，单击页面按钮等
- log目录：用于存放生成的日志文件

2、util包中新建一个名叫ObjectMap.py的Python文件

```
    用于实现定位页面元素的公共方法
from selenium.webdriver.support.wait import WebDriverWait
def getElement(driver,locateType,locatorExpression):
    try:
        element = WebDriverWait(driver, 30).until(lambda x: x.find_element(by=locateType,
value=locatorExpression))
        return element
    except Exception as e:
        raise e
# 获取多个相同页面元素对象，以list返回
def getElements(driver,locateType,locatorExpression):
    try:
        elements = WebDriverWait(driver, 30).until(lambda x: x.find_elements(by=locateType,
value=locatorExpression))
        return elements
    except Exception as e:
        raise e
```

3、在util包中新建WaitUtil.py文件，用于实现智能等待页面元素的出现

```
class WaitUtil(object):

    def __init__(self, driver):
    self.locationTypeDict = {
        "xpath": By.XPATH,
        "id": By.ID,
        "name": By.NAME,
        "css_selector": By.CSS_SELECTOR,
        "class_name": By.CLASS_NAME,
        "tag_name": By.TAG_NAME,
        "link_text": By.LINK_TEXT,
        "partial_link_text": By.PARTIAL_LINK_TEXT
     }
    self.driver = driver
    self.wait = WebDriverWait(self.driver, 30)
```

```
def frame_available_and_swith_to_it(self, locationType, locatorExpression):
    '''检查fram是否存在，存在则切换金frame控件中'''
    try:
        self.wait.until(EC.frame_to_be_available_and_switch_to_it
                ((self.locationTypeDict[locationType.lower()], locatorExpression)))
    except Exception as e:
        raise e
# 抛出异常信息给上层调用者

def visibility_element_located(self, locationType, locatorExpression):
    '''显示等待页面元素的出现'''
    try:
        element = self.wait.until(EC.visibility_of_element_located
                        ((self.locationTypeDict[locationType.lower()], locatorExpression)))
        return element
    except Exception as e:
        raise e
```

```
def visibilityOfElementLocated(self, locationType, locatorExpression, *arg):
    '''显示等待页面元素的出现'''
    try:
        element = self.wait.until(
            EC.visibility_of_element_located((
                self.locationTypeDict[locationType.lower()],
                locatorExpression)))
    except Exception as e:
        raise e
```

4、在testScripts包中新建一个名叫TestErpApprovalflow.py文件

```
/*用于编写具体的测试操作代码*/
def TestErpApprovalflow():
    driver = webdriver.Chrome()
    driver.maximize_window()  # 最大化浏览器
    driver.implicitly_wait(8)  # 设置隐式时间等待

    driver.get("http://10.1.2.58:8080/login.jsp")
    title = driver.title
    # if (title == "深圳市金斧子网络科技有限公司-ERP"):
    #     print("测试成功，结果和预期结果匹配！")
    print("输入登录用户名")
    # 输入用户名
    username = getElement(driver, "xpath", "//*[@name='username']")
    username.send_keys("defang2")
    print("输入登录密码")
    # 输入密码
    password = getElement(driver, "xpath", "//*[@name='password']")
    password.send_keys("123")
    print("点击登录按钮")
```

```
getElement(driver, "xpath", "//*[@class='login-btn']").click()

    assert "深圳市金斧子网络科技有限公司-ERP" in driver.title
    # print("标题"+ driver.title)
    print("标题" + driver.title)
    time.sleep(5)
    assert "德芳客服" in driver.page_source
    time.sleep(5)
    waitUtil = WaitUtil(driver)
    waitUtil.frame_available_and_swith_to_it("id", "iframe89892232323")
    print("点击审批流的链接")
    getElement(driver, "xpath",
"html/body/div[1]/div/div/div[2]/div/div/div/div[2]/div/div[1]/ul/li[1]/div[1]").click()
    time.sleep(3)
    print("切换到新的窗口")

    driver.switch_to.window(driver.window_handles[1])
    print("新窗口标题:", driver.title)
    time.sleep(3)
```

```
print("点击审批按钮")
    getElement(driver, "xpath", "//*[@id='btnAgree']").click()
    waitUtil.frame_available_and_swith_to_it("xpath", "//iframe[contains(@id,'ligerwindow')]")
    time.sleep(3)

    print("点击填写意见的确定按钮")
    getElement(driver, "xpath", "//*[@id='dataFormSave']").click()
    time.sleep(5)

    print("点击同意的确定按钮")
    getElement(driver, "xpath", "//*[text()='确定']").click()

    driver.quit()



if __name__ == '__main__':
    TestErpApprovalflow()
```

5、在config包中新建一个VarConfig.py的python文件

```
# 用于定义整个框架中所需要的一些全局常量值，方便维护

# 获取当前文件夹所在目录的父目录的绝对路径
parentDirPath = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
# 存放截图文件
screenPicturesDir = parentDirPath + "\\exceptionpictures"
```

6、在util包中新建一个名叫DirAndTime.py的python文件
用于获取当前日期及时间，以及创建异常截图存放目录

```
# 获取当前的日期
def getCurrentDate():
    timeTup = time.localtime()
    currentDate = str(timeTup.tm_year) + "-" + str(timeTup.tm_mon) + "-" + str(timeTup.tm_mday)
    return currentDate


# 获取当前的时间
def getCurrentTime():
    timeStr = datetime.datetime.now()
    nowTime = timeStr.strftime('%H-%M-%S-%f')
    return nowTime


# 创建截图存放的目录
def createCurrentDateDir():
    dirName = os.path.join(screenPicturesDir, getCurrentDate())
    if not os.path.exists(dirName):
        os.makedirs(dirName)
    return dirName
```

7、修改util包中的WaitUtil.py文件

```
class WaitUtil(object):

    def __init__(self, driver):
        self.locationTypeDict = {
            "xpath": By.XPATH,
            "id": By.ID,
            "name": By.NAME,
            "css_selector": By.CSS_SELECTOR,
            "class_name": By.CLASS_NAME,
            "tag_name": By.TAG_NAME,
            "link_text": By.LINK_TEXT,
            "partial_link_text": By.PARTIAL_LINK_TEXT
         }
        self.driver = driver
        self.wait = WebDriverWait(self.driver, 30)
```

```
def presence_Of_Element_Located(self, locatorMethod, locatorExpression, *arg):
        '''显式等待页面元素出现在DOM中，但并不一定课件，存在则返回该页面元素对象'''
        try:
            if self.locationTypeDict.has_key(locatorMethod.lower()):
                self.wait.until(EC.presence_of_all_elements_located((
                    self.locationTypeDict[locatorMethod.lower()], locatorExpression)))
            else:
                raise TypeError("未找到定位方式，请确认定位方式是否写正确")
        except Exception as e:
            raise e

    def frame_available_and_swith_to_it(self, locationType, locatorExpression, *arg):
        '''检查fram是否存在，存在则切换金frame控件中'''
        try:
            self.wait.until(EC.frame_to_be_available_and_switch_to_it
                        ((self.locationTypeDict[locationType.lower()], locatorExpression)))
        except Exception as e:
          # 抛出异常信息给上层调用者
            raise e
```

```
def visibility_element_located(self, locationType, locatorExpression):
        '''显示等待页面元素的出现'''
        try:
            element = self.wait.until(EC.visibility_of_element_located
                                    ((self.locationTypeDict[locationType.lower()], locatorExpression)))
            return element
        except Exception as e:
            raise e
```

8、在KeyWordsFrameWork工程中新建一个名叫action的python package，并在该包中新建PageAction.py,用于实现具体的页面动作，比如输入框输入数据，单击页面按钮等

```
driver = None
# 全局的等待类实例对象
waitUtil = None
def open_browser(browserName, *args):
    # 打开浏览器
    global driver, waitUtil
    try:
        if browserName.lower() == 'ie':
            driver = webdriver.Ie(executable_path=ieDriverFilePath)
        elif browserName.lower() == 'chrome':
            # 创建Chrome浏览器的一个options实例对象
            chrome_options = Options()
            # 添加屏蔽--ignore-certificate-errors提示信息的设置参数项
            chrome_options.add_experimental_option(
                "excludeSwitches",
                ["ignore-certificate-errors"])
            driver = webdriver.Chrome(
                executable_path=chromeDriverFilePath,
                chrome_options=chrome_options)
        else:
            driver = webdriver.Firefox(executable_path=firefoxDriverFilePath)
        # driver对象创建成果后，创建等待类实例对象
        waitUtil = waitUtil(driver)
    except Exception as e:
        raise e
```

```
def visit_url(url, *args):
    # 访问某个网址
    global driver
    try:
        driver.get(url)
    except Exception as e:
        raise e


def close_browser(*args):
    # 关闭浏览器
    global driver
    try:
        driver.quit()
    except Exception as e:
        raise e
```

```
def sleep(sleepSeconds, *args):
    # 强制等待
    try:
        time.sleep(int(sleepSeconds))
    except Exception as e:
        raise e
def clear(locationType, locatorExpression, *args):
    # 清除输入框内容
    global driver
    try:
        getElement(driver, locationType, locatorExpression).clear()
    except BaseException as e:
        raise e
def input_string(locationType, locatorExpression, inputContent):
    # 在页面输入框输入数据
    global driver
    try:
        getElement(driver, locationType, locatorExpression).send_keys(inputContent)
    except BaseException as e:
        raise e
```

```
def click(locationType, locatorExpression, *args):
    # 单击页面元素
    global driver
    try:
        getElement(driver, locationType, locatorExpression).click()
    except Exception as e:
        raise e


def assert_string_in_pagesource(assertString, *args):
    # 断言页面源码是否存在关键字或关键字字符串
    global driver
    try:
        assert assertString in driver.page_source, "%s not found in page source!" % assertString
    except AssertionError as e:
        raise AssertionError(e)
    except Exception as e:
        raise e
```

```
def getTitle(*args):
    # 获取页面标题
    global driver
    try:
        return driver.title
    except Exception as e:
        raise e


def getPageSoure(*args):
    # 获取页面源码
    global driver
    try:
        return driver.page_source
    except Exception as e:
        raise e
```

```
def switch_to_frame(locationType, frameLocationExpression, *args):
    # 切换进入frame
    global driver
    try:
        driver.switch_to_frame(driver, locationType, frameLocationExpression).click()
    except Exception as e:
        raise e
def switch_to_default_content(*args):
    # 切出fram
    global driver
    try:
        driver.switch_to_default_content()
    except Exception as e:
        raise e
def current_window_handle(*args):
    # 切换到新打开的页面
    global driver
    try:
             driver.switch_to.window(driver.window_handles[1])
             print("新窗口标题:", driver.title)
    except Exception as e:
        raise e
```

```
def maximize_browser():
    # 窗口最大化
    global driver
    try:
        driver.maximmize_window()
    except Exception as e:
        raise e


def capture_screen(*args):
    # 获取屏幕图片
    global driver
    curtTime = getCurrentTime()
    picNmaeAndPath = str(createCurrentDateDir()) + "\\" + str(curtTime) + ".png"
    try:
        driver.get_screenshot_as_file(picNmaeAndPath.replace('\\', r'\\'))
    except Exception as e:
        raise e
    else:
        return picNmaeAndPath
```

```
def wait_Presence_Of_Element_Located(locationType, locatorExpression, *args):
    '''显式等待页面元素出现在dom中，但并不一定可见,则返回改页面元素对象'''
    global driver
    try:
        waitUtil.presence_Of_Element_Located(locationType, locatorExpression, *args)
    except Exception as e:
        raise e
def wait_Frame_To_Be_Available_And_Switch_To_It(locationType, locatorExpression, *args):
    '''检查frame是否存在，存在则切换进frame控件中'''
    global waitUtil
    try:
        waitUtil.Frame_To_Be_Available_And_Switch_To_It(locationType, locatorExpression,
*args)
    except Exception as e:
        raise e
def waitVisibilityOfElementLocated(locationTpye, locatorExpression, *args):
    '''显式等待页面元素出现dom中，并且可见，存在返回改页面元素对象'''
    global waitUtil
    try:
        waitUtil.visibilityOfElementLocated(locationTpye, locatorExpression, *args)
    except Exception as e:
        raise e
```

9、修改testScripts包中的TestErpApprovalflow.py

```
def TestErpApprovalflow():
    print("启动Chrome浏览器")
    open_browser("chrome")
    # 最大化浏览器窗口
    maximize_browser()
    print("访问erp登录页面")
    visit_url("http://10.1.2.58:8080/login.jsp")
    time.sleep(3)
    assert_string_in_pagesource("深圳市金斧子网络科技有限公司-ERP")
    print("输入登录用户名")
    input_string("xpath", "//*[@name='username']", "defang2")
    print("输入登录密码")
    # 输入密码
    input_string("xpath", "//*[@name='password']", "123")
    print("点击登录按钮")
    click("xpath", "//*[@class='login-btn']")
    time.sleep(5)
    assert_string_in_pagesource("德芳客服")
    time.sleep(5)
    wait_Frame_To_Be_Available_And_Switch_To_It("id", "iframe89892232323")
    print("点击审批流的链接")
    click("xpath", "html/body/div[1]/div/div[2]/div/div/div/div[2]/div/div[1]/ul/li[1]/div[1]")
```

```
time.sleep(3)
    print("切换到新的窗口")

    current_window_handle()

    time.sleep(3)
    print("点击审批按钮")
    click("xpath", "//*[@id='btnAgree']")
    wait_Frame_To_Be_Available_And_Switch_To_It("xpath",
"//iframe[contains(@id,'ligerwindow')]")
    time.sleep(3)
    print("点击填写意见的确定按钮")
    click("xpath", "//*[@id='dataFormSave']")
    time.sleep(5)
    print("点击同意的确定按钮")
    click("xpath", "//*[text()='确定']")
    print("关闭浏览器")
    close_browser()


if __name__ == '__main__':
    TestErpApprovalflow()
```

作业：

1、完成上面的关键字驱动框架代码，并思考

Thanks!

科　技　提　升　投　资　品　质

金斧子
WWW.JFZ.COM