ScriptRunner 培训提纲

章节	标题	知识点
1	**Script Runner for Jira 介绍**	

1.1	Script Runner for Jira 概述	• 一款市场热销的 Jira 插件
		• 满足用户对 Jira 进行定制化轻量改造的需求
		  ○ 提供了多种类的保存和运行Groovy脚本的锚点
		  ○ 以内置脚本的形式提供了许多常用功能

1.2	对比插件开发与脚本开发	• 举例说明：当前用户为某固定用户时，允许用户转换流程（Condition）
		  ○ Scriptrunner: workflow - custom condition
		  ○ Scriptrunner: workflow - built-in script
		  ○ 环境安装 → 插件开发 → 插件安装 → 流程配置　相对繁琐

1.3	Script Runner 版本变更	• 版本之间会存在功能差别，请注意
		  ○ 早期版本的Condition vs 新版本中的Condition
		  ○ 逐步追加的功能 定时任务、工作流脚本汇总等

2	**Groovy 脚本的编写**	
2.1	内置变量	• 在不同的脚本设置界面点击问好图标查看支持的内置变量
2.2	Java API	• Jira API • ScriptRunner API

2.3	样例代码	• 脚本设置界面提供的样例代码
		• Document
		  ○ https://docs.adaptavist.com/sr4js/latest/get-started
		• Adaptavist Library
		  ○ https://library.adaptavist.com/

2.4	在IDE中带着提示写脚本	• Adaptvist官方教程
		  ○ https://docs.adaptavist.com/sr4js/latest/best-practices/write-code/set-up-a-dev-environment
		  ○ https://scriptrunner.adaptavist.com/latest/jira/DevEnvironment.html
		• 插件开发项目中编写
		  ○ idea配置：Project Structure → Modules → Add → Groovy 。 Groovy编译器下载 http://www.groovy-lang.org/

2.5	一些 Groovy 语法适用的常见写法	• def • ? • [].any{ it -> /* */ } • [].each{ it -> /* */ }
3	**Browse ScriptRunner**	
3.1	分类查看ScriptRunner功能	• 管理员内置工具操作 ADMINISTOR • 自动化处理 AUTOMATE • 自定义功能 CUSTOMISE • 其它 EXTEND • Categories：Project , Issue , Fields , User , Reporting , System , Workflow , UI , Email

4	**Console & Script Editor**	

4.1	Script Console	• 脚本运行窗口，作用:
		  ○ 运行一次性脚本，例如数据处理
		  ○ 测试验证Jira API
		• 两种添加脚本的方式:
		  ○ Script
		  ○ File
		• 脚本窗口缩放
		• 运行结果查看
		  ○ Result
		  ○ Logs
		  ○ Timing

4.2	Script Editor	• 创建、编辑和保存 Groovy 脚本文件 • 以文件目录的形式管理脚本文件 • 可以被各种脚本控制台引用
5	**Built-in Scripts**	
5.1	Bulk Fix Resolutions	• 批量修改问题的解决结果 ○ 指定一个问题筛选器，指定新的解决结果，执行。
5.2	Bulk import custom field values	• 为选择类型的字段导入选项 ○ 选择要导入选项的字段配置方案，填入要导入的选项，执行。
5.3	Change dashboard or filter ownership	• 变更筛选器Filters和仪表板Dashboards的负责人 ○ 选择原用户和目标用户，选择要变更的仪表板和筛选器，执行。
5.4	Clean Workflows	• 找出工作流中装载的由已卸载/禁用插件提供的功能，包括后处理、条件、验证器等，支持清除

5.5	Copy field values	• 批量修改问题，把一个字段值复制给另一个字段，字段类型变更迁移历史数据时常用
		  ○ 指定一个问题筛选器，指定原字段，指定新字段，执行
		  ○ 选择类型的字段值以选项值为准（不是选项ID）

5.6	Copy project	• 复制一个项目，包括项目的方案、角色成员、字段设置等，可选同时复制项目中的问题、版本、模块、请求类型（JSM）、队列（JSM）、组织（JSM）、SLAs（JSM）
5.7	Service Desk Template Comments (JSM)	• 设定服务台项目评论回复时可选的评论模板
5.8	Generate events	• 发射事件 - 为指定问题筛选器或项目中的所有问题抛出指定的Event
5.9	Re-index Issues	• 为指定问题筛选器或项目中的所有问题重建索引
5.10	Script registry	• 查看所有ScriptRunner中注册的自定义脚本，帮助快速找到有问题的脚本代码
5.11	Split custom field contexts	• 拆分字段的配置方案，使指定项目、问题类型应用新一套相同内容（选项、默认值）的配置方案，问题中的字段值也变为新选项值。

5.12	Switch to a different user	• 扮演指定用户身份
5.13	List scheduled jobs	• 列出系统中所有的定时任务
5.14	View server log files	• 查看服务器日志文件【常用】
6	**Listeners**	

6.1	Custom listener	• 自定义Groovy脚本配置监听动作
		  ○ 项目作用域
		  ○ 事件选择（根据名称猜测事件）
		  ○ 捕捉到事件后执行的脚本
		• 实践：添加issueUpdateEvent监听并验证
		`if(event.issue.priority?.name == 'High' && event.changeLog?.getRelated ("ChildChangeItem").any { it.get('field')=='priority' }){ log.warn("Priority changed to High!") }`

6.2	Remote custom listener / listener dispatcher	• 使建立了应用程序链接的系统实例之间可以相互抛送事件，实现跨系统监听
6.3	Send a custom email (non-issue events)	• 事件触发发送自定义邮件（非issue相关事件）
6.4	Version synchroniser	• 一个项目中版本的创建、编辑、发布等动作，都会同步到其它项目执行
6.5	Adds the current user as a watcher	• 把当前触发事件的用户添加为问题的关注人，可以用脚本附加条件

6.6	Clones an issue, and links	• 复制当前事件对应的问题，可以复制为其它的项目、问题类型
		  ○ 复制得到的问题与当前问题之间建立问题链接
		  ○ 可以定义复制的字段范围、是否复制评论、子任务
		  ○ 可以设置触发复制的限定条件（脚本）
		  ○ 可以设置复制执行过程的追加动作（脚本），如设置目标issue字段值等

6.7	Create a sub-task	• 为当前事件对应的问题创建一条子任务，设置内容与clone类似

6.8	Fast-track transition an issue	• 转换前事件对应的问题的工作流状态
		  ○ 可以设置需要满足的条件（脚本）
		  ○ 转换相关的设置：选择动作、Skip Permissions、Skip Validators、Skip Conditions
		  ○ 可以设置复制执行过程的追加动作（脚本），如设置issue字段值等

6.9	Fires an event when condition is true	• 给事件的抛出追加条件（脚本），如“优先级为High”
6.10	Post a message to Slack	• 事件触发向Slack频道发送消息

6.11	Send a custom email	• 事件触发发送电子邮件
7	**Behaviours**	
7.1	Behaviours的作用	• 仅作用于 Jira 弹窗界面中，主要是问题的创建、编辑、转换窗口。在非弹窗模式打开的创建界面里无效。 • 可以控制窗口交互过程中的界面变化。
7.2	Behaviour Mappings	• 此下的脚本和设置会生效的作用域，按项目、问题类型设定，必须设定。

7.3	Initialiser Function	• 仅在窗口打开时执行一次的脚本
		• 实践：添加 Initialiser Function 并验证
		`/**  */`
		`getFieldById("duedate").setRequired(true)`
		`/**  */`
		`getFieldById("attachment").setHidden(true)`
		`/**  */`
		`getFieldById("duedate").setDescription("")`
		`/**  */`
		`getFieldById("description").setFormValue("demo")`

7.4	字段的必填、只读、隐藏设置	• 实践：分别设置并验证

7.5	Server-side script for fields	• 在窗口打开时执行一次，且关联的字段每次值发生变化时也会执行一次的脚本
		• 实践：添加 Server-side script 并验证
		`import com.atlassian.jira.issue.priority.Priority`
		
		`def priority = getFieldById(getFieldChanged()).value as Priority`
		`if(priority){`
		`  getFieldById("description").setDescription(priority.name)`
		`}`

7.6	conditions of behaviour for field	• 可以控制字段配置生效的条件 • 条件对必填、只读、隐藏设置有效，对 Server-side script 可能没有限制作用（可能跟ScriptRunner版本有关，需持续验证）
7.7	underlyingIssue	• 窗口打开之前的issue对象，常用于对比用户操作前后变化的字段内容。如果是创建Issue的窗口，该对象不存在。
8	**Workflows**	

8.1	Jira 工作流机制	• 工作流的组成
		  ◦ 状态
		  ◦ 转换
		• 转换对应Issue界面中的操作按钮
		• 转换过程可以关联界面
		• 可以设置的转换要素
		  ◦ conditions 条件
		  ◦ validators 验证器
		  ◦ post-functions 后处理
		  ▪ 原生的工作流后处理
		  ▪ 注意后处理的顺序
		• 工作流编辑完成需要发布
		• 工作流方案 - 任务类型与工作流的映射表
		  ◦ 方案的复制、配置与引用

8.2	Custom script condition	• 实践：在工作流中添加一个 Custom script condition 查看效果
		`/**  */`
		`passesCondition = Objects.nonNull(issue.assignee)`

8.3	Custom script validator	• 实践：在工作流中添加一个 Custom script validator 查看效果
		`import com.opensymphony.workflow.InvalidInputException`
		
		`if(Objects.isNull(issue.getDescription())){`
		`  throw new InvalidInputException("")`
		`}`

8.4	Custom script post-function	• 实践：在工作流中添加一个 Custom script post-function 查看效果 `issue.setDescription(issue.getSummary() + " - " + issue.getAssignee())`
8.5	内置的 conditions	• 对照插件添加 condition 的界面逐一浏览一遍
8.6	内置的 validators	• 对照插件添加 validator 界面逐一浏览一遍
8.7	内置的 post-functions	• 对照插件添加 post-function 界面逐一浏览一遍
9	**Jobs**	

9.1	Custom scheduled job	• 定时执行脚本
		  ◦ 支持按间隔或Cron表达式设定时间
		  ◦ 支持指定脚本执行使用的用户身份
		• 实践：添加 job 并验证
		`import com.atlassian.jira.component.ComponentAccessor`
		`def issueService = ComponentAccessor.issueService`
		`def user = ComponentAccessor.jiraAuthenticationContext.getLoggedInUser()`
		`/** issue*/`
		`def issue = issueService.getIssue(user, "DEMO-1").getIssue()`
		`def issueInputParameters = issueService.newIssueInputParameters()`
		`issueInputParameters.setDescription(user.username)`
		`def result = issueService.validateUpdate(user, issue.getId(), issueInputParameters)`
		`if (!result.getErrorCollection().hasAnyErrors()) {`
		`  issueService.update(user, result)`
		`}`

9.2	内置的 jobs	• Escalation service - 定期处理符合条件issue，转换工作流状态、改变字段值、添加评论等 • Issue archiving job - 定期归档符合条件的issue
10	**Fields**	
10.1	ScriptRunner Fields 概述	• 每新建一个 Script Field 都是在系统中创建了一个自定义字段 • 字段的类型是由 ScriptRunner 插件扩展的，每种类型具有不同的效果 • Script Field 字段既拥有 ScriptRunner 的配置，也拥有 Jira 自定义字段的配置

10.2	Custom Script Field	• 一个实时计算并变化值的字段
		  ◦ 每次尝试获取字段值时都会执行计算，不论是打开问题查看界面、还是出现在问题查询、导出显示的结果中
		  ◦ 因为执行场景过多，务必注意不要放入过于耗时的脚本，如网络请求、批量处理、大范围查询等
		  ◦ 务必设置好字段的作用域 Contexts，无关项目避免执行脚本
		  ◦ 创建时注意选择正确的 Template 呈现结果数据
		• 字段值不支持编辑
		• 字段不支持在JQL中作为条件搜索
		• 适用的场景
		  ◦ 根据问题中已有的信息或者关联的问题信息，自动计算出结果用于显示
		• 实践：添加 Custom Script Field 并验证
		`import com.atlassian.jira.component.ComponentAccessor`
		`def attachmentManager = ComponentAccessor.getAttachmentManager()`
		`def num = attachmentManager.getAttachments(issue).size()`
		`return num ? num as Double : null`

10.3	Issue(s) picker	• 问题选择器
		  ◦ 指定一个JQL，查询到的问题会作为这个字段的选项
		  ◦ 可单选，可多选

10.4	Custom Picker	• 自定义选项的选择器
		  ◦ 自己用脚本获取并控制返回字段的候选项，比如调用一个接口获取
		  ◦ 可单选，可多选

11	**Fragments**	

11.1	Custom web item	• 添加链接按钮
11.2	Constrained create issue dialog	• 添加按钮，点击按钮打开指定项目、问题类型的创建问题窗口
11.3	Show a web panel	• 添加 Web Panel 面板
11.4	Create a custom web section	• 添加自定义 Web Section
11.5	Hide system or plugin UI element	• 隐藏某些 UI 元素
12	**REST Endpoints**	

12.1	Custom endpoint	• 实践：添加一个 Custom endpoint 并调用 /rest/scriptrunner/latest/custom/doSomething 验证 （示例来自 Show snippets）
		`import com.onresolve.scriptrunner.runner.rest.common.CustomEndpointDelegate`
		`import groovy.json.JsonBuilder`
		`import groovy.transform.BaseScript`
		
		`import javax.ws.rs.core.MultivaluedMap`
		`import javax.ws.rs.core.Response`
		
		`@BaseScript CustomEndpointDelegate delegate`
		
		`doSomething(httpMethod: "GET", groups: ["jira-administrators"]) { MultivaluedMap queryParams, String body ->`
		`  return Response.ok(new JsonBuilder([abc: 42]).toString()).build();`
		`}`

13	**JQL Functions**	
13.1	ScriptRunner对 JQL 的扩展支持	• 示例：某个Epic下的所有标准任务以及其下的所有子任务 ◦ key = DEV-1436 OR "Epic Link" = DEV-1436 OR issueFunction in subtasksOf('"Epic Link" = DEV-1436')
14	**Resources**	
14.1	Database connection	• 配置数据库连接，供其它脚本使用 ◦ 参考文档：https://docs.adaptavist.com/sr4js/6.39.0/features/resources/database-connection?utm_source=product-help
15	**Jira 常用 Java API**	
15.1	JAVA DOC	• 通过Atlassian Developer网站进入 ◦ https://docs.atlassian.com/software/jira/docs/api/latest/
15.2	用户与组管理	• 实践：创建一个用户组的接口 • ComponentAccessor • JiraAuthenticationContext • UserManager • GroupManager

15.3	Issue的创建、更新与删除	• 实践：创建一个issue的接口 • 实践：更新issue的summary和description的接口 • 实践：删除一个issue的接口 • IssueManager • MutableIssue • EventDispatchOption

15.4	自定义字段值读写	• CustomFieldManager实践：在创建issue时为自定义字段赋值
		  ○ 调整创建Issue方法，在创建时传入自定义字段值的参数
		  ○ 使用CustomFieldManager获取自定义字段对象，设置自定义字段的值
		  ○ 调用创建Issue的接口，查看创建结果中对应字段的值

15.5	其它常用管理与工具类	• WorklogManager常用方法
		• ChangeHistoryManager常用方法
		• WorkflowManager&IssueService实践：转换issue状态
		  ○ 使用WorkflowManager根据Issue对象获取关联的工作流，从而可以获取工作流转换动作和状态等信息，取到动作id
		  ○ 使用IssueService执行指定的转换动作
		  ○ 实现测试接口，演示调用
		• SearchService查询Issue

16	**使用SctiptRunner需要注意的问题**	
16.1	注意规避性能问题	• 如果需要批量处理大量数据，比如查询Issue并操作，请尽量控制处理范围，优化过程，并预估数据增量充分测试。批量创建、修改Issue会伴随较大的系统资源占用，严重时可能导致宕机。 • 涉及批量数据操作的定时任务注意避开高峰时段，Job之间也要错峰。 • Listeners 或 Post-Functions 中出现耗时操作，例如外发请求、大批量数据查询处理等，可能造成用户UI界面卡顿甚至超时异常。如不可优化或避免，可以开启新线程执行。
16.2	脚本的作用域要严格控制	• Behaviours 的 Mapping • Fields 的 Contexts • Listeners 的 Projects & Events
16.3	脚本统一管理要求	• 经过严格测试/审核再上线运行 • 标题和表述要准确 • 尽可能遵守一些脚本编写的规范

16.4	脚本编写规范	• 脚本注释说明
		  ○ 脚本类型、安装位置
		  ○ 安装过程、相关配置步骤
		  ○ 脚本业务负责人、开发者
		  ○ 脚本最后更新时间
		• 日志输出
		  ○ 注意控制log级别，避免正常执行时输出大量冗余日志，避免异常时无法定位到具体脚本。

16.5	适度安装脚本	• 脚本虽好用，安装请适度，脚本装越多，系统越笨重。