一、发现页面的导入规则解析逻辑

书源作为爬虫的解析规则，支持 CSS 选择器、JSON、JSON-PATH、Xpath、正则表达式规则（下面有讲具体写法）。

1.无论小说网页网址，属性规则都是有规律的。

2.网址寻找有发现规律的字段规则，写入自定义规则展示中，以丰富自定义规则的展示内容。

3.这些规则基于书源的 HTML 内容或者 JSON 响应，依赖于书源网站的结构，随网站结构的更新而更新。

用户交互

书源管理界面

界面元素：添加书源按钮、书源列表（编辑/删除）、每个书源的基本信息展示。

功能：添加书源按钮、书源列表（编辑/删除）、每个书源的基本信息展示。

书源导入界面

界面元素：文件选择器、导入按钮、格式说明文本（帮助用户了解需要的 JSON 格式）。

功能：用户通过文件选择器上传书源文件，通过“导入”按钮确认。如果格式错误，系统会显示错误信息和正确格式的提示。

自定义规则编辑器界面

界面元素：规则字段输入框、说明文本、保存/测试规则按钮。

功能：高级用户可以编辑自定义的抓取规则，保存测试规则的有效性。

1. 书源文件识别规则导入

- **规则目标：**能够识别和解析书源文件导入。
- **验证方法：**自动读取用户导入的书源文件，尝试解析 JSON 数据。如果数据符合预期的结构和数据类型，认为书源文件有效。如果存在任何格式错误和缺少必要的字段，系统将拒绝文件给出相应的错误提示。
- **信息展示：**如果书源文件通过验证，提取文件中的关键信息（如书源的名称、对应的网址 URL）在界面上展示给用户看，让用户确认所导入的书源信息。

2. 书源内容解析规则

- **规则目标：**从书源网页中抓取所需信息:书名、作者、书籍详情、简介、标签、章节列表、正文各等，下面有讲。
- **数据清洗：**对抓取的数据进行必要的清洗，包括去除广告文本、不必要的 HTML 标签、空白符。

3. 错误处理规则

- **规则目标：**处理无效书源、网络错误、数据解析失败。
- **处理方法**
  - **无效书源：**如果书源 URL 不可访问或不包含书籍信息，记录错误信息，通知用户。
  - **网络错误：**如果抓取过程中发生网络错误（超时、失败），进行重试和记录错误详情。
  - **解析失败：**如果无法从页面中抓取预期信息，记录错误详情，标记书源为“失败”。

4. 书源信息展示规则

- **规则目标：**在用户界面上清晰、准确地展示书籍信息。
- **展示方法**
  - **书籍列表：**书源多本书以列表形式展示，每项包含书名、作者等基本信息。
  - **章节列表：**用户选择一本书后，展示章节列表。
  - **搜索引擎：**精准搜索书名和作者。
  - **正文展示：**用户选择任何章节后，以合适的格式展示正文。

5. 用户编写自定义规则编辑器（中级和高级）

- **规则目标：允许用户根据个人喜好特定需求，自定义抓取和展示规则。**
- **展示方法**
  - **自定义选择器：**用户输入自定义的 Json 、Res、XPath 改变抓取逻辑。
  - **自定义数据处理：**设置自定义规则实现.

书源主要由 6 部分组成

6.1 基本信息 是否会抛出异常，提示缺少填写必要字段

（1）书源 URL

- 描述：书源的网址，是唯一标识，不可重复。
- 格式：URL
- 示例： https://m.81zw.so/

（2）书源名称

- 描述：书源的名称，用于在列表显示。
- 格式：字符串，描述书源名称。
- 示例：“八一中文网”。

（3）请求头（非必填）

- 描述：在游览器发出的 HTTP 请求一起发送的一组键值对信息。
- 格式：请求头通常是“名称: 值”对的格式，每对之间用换行符分隔，HTTP 头区分大小写。
- 示例：

```
{
    “User-Agent”：xxxxx
    “Accept-Language”:xxxxx
}
```

（4）发现 URL

- 描述：用于列出分类索引的书籍、热门阅读列表或特定类别/标签下的作品。
- 格式：URL 可能包含用于筛选书籍的查询参数。
- 示例：{ “tag_thriller”： “http://example-bookshelf.com/tag/玄幻” }

（5）搜索 URL

- 描述：搜索 URL 向书源网站发起搜索请求的基础。说不定 URL 是动态生成的，根据用户输入的搜索关键词和特定书源网站的查询。
- 格式：协议://域名[:端口号]/ ?查询参数。
- 示例：`http://example.com/search?keyword={keyword}`，{keyword}用户搜索书名关键字。

6.2 全局参数

一、搜索规则

(1)书籍列表规则(BookListRule)

- 描述：定义如何从搜索结果 HTML 中定位书籍列表的 JSON 和正则表达式。
- 格式：JSON 和正则表达式。
- 示例：`$.data.*、`result.books.*`

(2)书名规则(NameRule)

- 描述：从每个书籍条目中提取书名的规则。
- 格式：JSON 和正则表达式，指向书名。
- 示例：`$.name` 、`bookName`

(3)作者规则(AuthorRule)

- 描述：从书籍信息中提取作者名的规则。
- 格式：JSON 和正则表达式，指向作者名。
- 示例：`$.author`、`auhtorName`

(4)分类规则(KindRule)

- 描述：用于确定书籍所属的类别和分类。
- 格式：JSON 和正则表达式，指向书籍分类。
- 示例：`$.category`、`genre`

(5)字数规则(WordCountRule)

- 描述：从书籍信息中提取总字数的规则
- 格式：JSON 和正则表达式，指向字数信息。
- 示例：`$.wordCount`、`totalWords`

(6)最新章节规则(LastChapterRule)

- 描述：从书籍信息中提取最新章节标题的规则。
- 格式：JSON 和正则表达式，指向最新章节信息。
- 示例：`$.lastChapter`、`latestChapter`

(7)简介规则(BriefRule)

- 描述：从书籍信息中提取简介的规则。
- 格式：JSON 和正则表达式，指向书籍的简介。
- 示例：`$.introduction`、`summary`

(8)详情页规则(DetailUrlRule)

- 描述：从书籍信息中提取详情页面 URL 的规则。
- 格式：JSON 和正则表达式，指向书籍详情页 URL。
- 示例：`$.link`、`detailUrl`，"detailUrl":\s*"([^"]+)"

二、发现规则

书籍列表规则(BookListRule)

书名规则(NameRule)

作者规则(AuthorRule)

分类规则(KindRule)

字数规则(WordCountRule)

最新章节规则(LastChapterRule)

- 描述：从书籍信息中提取最新章节标题的规则。
- 格式：JSON 和正则表达式，指向最新章节信息。
- 示例：`$.lastChapter `、`latestChapter`

简介规则(BriefRule)

详情页规则(DetailUrlRule)

三、详情规则

预处理规则（bookInfoInit）

- 描述：用于初始化书籍的元数据和内容，包括验证、清洗、数据转换和合并收集到的信息。
- 格式：用 JS 和 Res 脚本代码，处理用户脚本。
- 示例：function bookInInit(bookInfo) {`[^>]+>/g`}

此函数 bookInfoInit 接受一个原始的 bookInfo 对象，执行一系列预处理步骤来改进和准备数据。这些步骤包括验证必要的字段（如标题和作者）、清理从原始源中提取的数据（如去除描述中的 HTML）、添加缺少的信息（如默认封面图像）。

四、目录规则

目录列表规则(ChapterListRule)

- 描述：规则用于识别和抽取特定书籍的全部章节列表，指向章节条目的页面或页面部分。
- 格式：JSON 和正则表达式，指向章节列表的容器元素。
- 示例：`$.data.chapters[*]`、`<li class="chapter">(.*?)</li>`

章节名称规则(ChapterNameRule)

- 描述：从章节列表的每个条目中提取具体的章节名称。
- 格式：JSON 和正则表达式,针对每个章节条目的名称
- 示例：`$.title`、`<a .*?>(.*?)</a>`

更新时间（UpdateTimeRule）

- 描述：从书籍或章节元数据中抽取最后更新时间。
- 格式：JSON 和正则表达式，定位表示时间戳或日期时间的字段。
- 示例：`$.updateTime`、`<span @class>`

五、正文规则

正文规则(ContentRule)

- 描述：从网页中准确提取章节的全文内容，避免抓取到不必要的信息，比如导航链接、广告或者其他非正文的元素。
- 格式：JSON 和正则表达式，解析页面结构。
- 示例：`$.content`、`(?<=<div[^>]*>)(?s)(.*?)(?=<\/div>)`

六、其他高级规则

（1）登录 URL(LoginUrl)

- 描述：有些小说网要求先登录后才能看书架上的书籍。
- 格式：协议://域名[:端口号]/
- 示例：`loginUrl`:`https://xxxxx/login`

(2)登录参数规则(LoginParamsRule)

- 描述：填参数。
- 格式:username、password
- 示例：

```
`{
`proxy_type`: `HTTP`,
`proxy_address`: `http://proxyserver:port`,
`username`: `xxx`,
`password`: `xxx`
}`
```

(3)用户代理规则（ProxyRule）

- 描述：模仿特定类型的网络请求。
- 参数：user_agent_string: 设置的具体用户代理字符串。
- 示例：

```
{
        `rule_type`: `UserAgentRule`,
        `user_agent_string`: `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537`
}
```

(4)Cookie 规则(CookiesHandlingRule)

- 描述：处理服务器发送的 Cookie 以及客户端返回给服务器的 Cookie。
- 格式：获得用户的同意才能存储 Cookie。
- 示例：`$.headers['Set-Cookie']`、`(?<=Set-Cookie:\s)(.*?)(?=;)`

(5)超时设置(TimeoutSettings)

- 功能描述：允许用户自定义规则设置操作最大时长，等待设备和服务器建立连接的最长时间。

- 格式：%秒
- 示例：连接超时：[30] 秒
  当用户按下“保存设置”按钮时，应用将验证并应用新的超时设置。
  如果操作超出用户定义的时限，应中断该操作，显示一个错误消息。例如：“操作超时，请检查您的网络连接，尝试调整超时设置。”

调试模块功能

(1)搜索调试：
- 用户通过输入书名或作者，测试是否能够返回正确的结果。

(2)发现 URL 调试：
- 测试发现 URL 是否有效，列出最新的书籍列表。

(3)详情页 URL 调试：
- 测试详情页 URL 规则是否正确，显示规则抓取到书籍信息。

(4)目录页 URL 调试：
- 确保目录页 URL 能正确展示书籍的各个章节列表。

(5)正文页 URL 调试：
- 验证正文页 URL 规则的有效性，能抓取文本内容。

(6)实时日志调试
- 全局记录调试过程。

4.获取网络书籍（WebBook），如果无法联网不能解析规则导入。

1. 获取书籍列表的方法

核心逻辑：

- 发起搜索请求：输入网址后提取想要的规则展示结果。

2. 展示书籍列表：提取的书籍信息整理为标准格式，在界面以列表的形式展示给用户看。

3. 获取书籍详情的方法

4. 获取目录的方法

5. 获取正文的方法

异常处理：

- 网络不稳定，提示告诉用户请重新刷新。

放入书架模块的流程

功能描述：

允许用户把书籍内容缓存到本地存储器，在无网络连接的情况下享受阅读。通过使用多线程并发下载，能够在短时间内处理大量下载任务，提升性能和响应速度。

下载队列管理

(1)用户可以添加书籍到下载队列。

(2)如果队列中的任务数量超出限制，新的任务将等待之前的任务完成。
(3)用户可以取消正在队列中等待的下载任务。

离线缓存
(1)下载完成的书籍保存在本地存储器中。
(2)用户可以在无网络连接的情况下阅读已缓存的书籍。
(3)使用多线程下载技术，允许多个下载任务同时进行。
每个下载任务分配给一个单独的线程，确保下载过程的流畅性和稳定性。

异常处理
(1)在下载过程中出现获取错误的时候，请告知用户尝试重新下载。
(2)如果下载任务失败，告诉用户收到通知。

并发控制
(1)使用线程池管理并发下载，控制最大线程数量，防止内存资源过度使用。
(2)实现线程同步，确保数据的一致性和完整性。

用户界面
(1)提供一个清晰的界面显示当前的下载队列。
()2 允许用户管理（暂停、取消、重新开始）下载任务。

性能考虑
(1)平衡并发下载的数量，避免过多的线程消耗过多资源。
(2)下载的文件进行有效优化，减少网络使用和存储空间。

异常处理
(1)如网络中断、文件损坏、存储空间不足。
(2)提供错误日志记录功能，分析问题发生的原因。

高级功能
(1)如限制下载速度、定时下载、仅在 Wi-Fi 环境下自动下载。

告诉用户引导教程：
正则语法：
支持分组选择，采用 ## 分割正则与分组，$1 代表第一组，$2 代表第二组，类推。
// 对应正则规则
regex:(.*?)年(.*?)月##$1-$2
JsonPath 规则
基本用法如下：

```
{
```

```
    "store": {
        "book": [
            {
                `category`: `xxx`,
                `author`: `xxx`,
                `title`: `xxx`,
            },
        ],
    },
}

// 规则
$.store.book[0].author
```

Xpathg 规则：
写法举例
```
//*[@id="app"]/div[1]/aside/ul/li[1]/section/ul/li[2]/a@href
```

搜索参数的位置
可以定义关键词与页码参数。
关键词：{{keyword}} 、页码：{{page}}
对于 GET 请求：
https://xxx.com?page={{page}}&kw={{keyword}}
https://xxx.com/search {{keyword}}/{{page}}
对于 POST 表单或者 JSON 的，参数放入请求体中。
```
{
    `body`: `page={{page}}&kw={{keyword}}`
}
```
如果填写了请求参数中的编码字段，采用对应编码对搜索关键词进行 unicode 编码。
如果详情页面规则未填写则默认从搜索结果页(search)匹配
```
`url`: {
    `rule`: `xpath:/body/div[3]/ul`,
    `page`: `search/detail`,
}
```
进阶写法：

二、阅读体验

1.文本预处理

（1）数据清洗：去除原始文本中的**字符**或**和内容无关**的格式信息，只保留文本内容，内容无关的信息处理：html 标签、Unicode 字符。

（2）标记化：纯文本内容分解为更小的单位：段落。
（3）结构识别：识别文本中的结构元素：标题、子标题、正文、句尾。

技术选择

1、使用 JSOUP 解析 HTML 提取纯文本。
2、正则表达式用于清除处理，例如删除多余的空格和非打印字符。
3、确保解析、存储和显示过程中文本的字符编码保持一致，比如设备上的兼容性（UTF-8），避免乱码文本。
4、OKHTTP 用于网络请求，获取原始数据。
5、Split 库处理文本进行有效的标记化。

2. 节点元素绘制

节点定义：文本节点和图像节点分别对应它们的内容。
布局属性：位置、文字颜色、字体样式（仿宋、黑体、微软雅黑）、粗体\细体、字体排期。
交互：实现内容的各种手势控制，如点击、滑动、长按。
技术选择
自定义 View 和 ViewGroup 实现特定的布局属性和交互。
GestureDetector 类用于处理复杂的手势交互。

3. 渲染优化

性能优化：使用算法和数据结构渲染过程的性能。
动态加载：节省内存和 CPU，避免发热，包括动态加载和卸载。
实现资源的按需加载和释放。
适应调整：根据用户的设备特性和偏好设置，动态调整节点元素的布局和样式。

4. 阅读功能

1.听书模式：TTS 朗读正在播放，定时设置分别是 0、15 分钟、30 分钟、45 分钟和 60 分钟。
2.自动阅读模式：页面自动滚动，无需要手动翻页。
3.智能摘要模式：使用 AI 快速生成章节和书籍的摘要，帮助用户快速回顾，是否继续阅读。
4.沉浸式阅读模式：

5. 智能调整

一、智能自动亮度：根据当前时间、用户的位置，外光线会自动调整亮度，减少眼镜疲劳。（通过访问设备的光线传感器）
二、智能主题切换：根据一天的不同时间，以当前环境的方式展现。
（1）日间模式：利用设备的时钟和地理位置服务确定日出时间，反映白天的光线条件。
（2）夜间模式：利用设备的时钟和地理位置服务确定日落时间，根据应用程序系统的变化将启动夜间。

6. 书籍信息模块

1. 封面图像

界面元素：在显示区域上应用程序有一个明显的区域显示书籍的封面图像，大小应适中，清晰展示封面图像。

功能：

可以点击封面图像，进入全屏模式查看，还可以保存。

界面元素：显示一组预定义的标签，如“科幻”、“爱情”、“冒险”等，一个允许用户添加自定义标签的选项。

功能：用户可以快速了解书籍的主题和特点，通过标签类似兴趣的用户互动。还有助于个性化推荐和精确的搜索结果。

2. 放入书架

   界面元素：一个明确的按钮图标，表明点击当前书籍添加到他们的书架中。

   功能：用户可以方便地保存他们感兴趣的书籍。

3. 开始阅读

   界面元素：一个突出显示的按钮，引导用户开始阅读书籍。

   功能：这个按钮提供快速入口，直接进入阅读界面。

4. 刷新

   界面元素：一个刷新图标，位于界面的顶部。

   功能：如果书籍信息有更新（例如新章节），可以点击此按钮刷新内容。

5. 下载阅读

   提供一个选项下载书籍，离线阅读。