Асинхронные запросы в HttpClient

JSP & Servlets
10 уровень, 4 лекция

ОТКРЫТА

5.1 Метод sendAsync()

Также с помощью HttpClient'а можно посылать асинхронные запросы. Обычно это делают в трех случаях.

Первый случай – это **запрос будет выполняться очень долго**, например, отправка/получение файла. Тогда эту операцию запускают и выполняют асинхронно.

Второй случай – тебе **нужно отправлять запросы очень часто**, и ты не хочешь ждать ответа от предыдущего запроса перед отправкой следующего.

И наконец, третий случай – тебе **не важен результат твоего запроса**. Например, ты раз в минуту делаешь скриншот своего экрана и отправляешь его на сервер. То есть логика твоего приложения предполагает, что запросов много и доходят не все. Тогда удобно работать по принципу – отправил и забыл.

Для того, чтобы отправить асинхронный запрос, нужно вызвать метод `sendAsync()` у объекта класса HttpClient. Этот метод мгновенно завершает работу и возвращает объект `CompletableFuture<HttpResponse>`. С его помощью можно отследить, когда запрос реально выполнится, а также выполнить определенный код после завершения запроса. Пример:

```
1  HttpClient client = HttpClient.newBuilder().build();
2
3  CompletableFuture<HttpResponse<String>> response = client.sendAsync(
4          request,
5          HttpResponse.BodyHandlers.ofString()
6  );
```

Метод `sendAsync()` возвращает объект `CompletableFuture`, который внутри себя содержит HttpResponse, который внутри себя содержит строку, которую вернет сервер.

5.2 Метод executor(), ExecutorService

Также HttpClient позволяет передать в него `ExecutorService` (пул потоков), которые будут использоваться для выполнения асинхронных запросов. Собственно, в серверных Java-приложениях так всегда и делают.

Ведь если на каждый входящий запрос к твоему API, ты будешь запускать несколько асинхронных запросов еще куда-то, вам никаких потоков не хватит. Пример:

```
1  ExecutorService executorService = Executors.newFixedThreadPool(2);
2
3  CompletableFuture<HttpResponse<String>> response1 = HttpClient.newBuilder()
4    .executor(executorService)
5    .build()
6    .sendAsync(request, HttpResponse.BodyHandlers.ofString());
7
8  CompletableFuture<HttpResponse<String>> response2 = HttpClient.newBuilder()
9    .executor(executorService)
10   .build()
11   .sendAsync(request, HttpResponse.BodyHandlers.ofString());
```

Если пул потоков не задан, то по умолчанию используется `java.util.concurrent.Executors.newCachedThreadPool()`.

‹ Предыдущая лекция

Следующая лекция ›

JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

Русский

СКАЧИВАЙТЕ НАШИ ПРИЛОЖЕНИЯ

ДОСТУПНО В Google Play

Загрузите в App Store