12

Implementando requisição para inserção de aluno

Agora que criamos a AsyncTask e o conversor completo para o aluno, precisamos de fato criar o método que enviar a requisição de inserção. Para isso, inicialmente faremos a chamada da classe `WebClient` dentro do `doInBackground` da `InsereAlunoTask`:

```
@Override
protected Object doInBackground(Object[] params) {
    String json = new AlunoConverter().converteParaJsonCompleto(aluno);
    new WebClient();
    return null;
}
```

Em seguida, vamos adicionar o método `insere()` enviando o `json` como parâmetro nesse mesmo trecho de código. Então, pressione o **Alt + Enter** e peça para que o Android Studio crie o método automaticamente, então teremos o seguinte resutaldo:

```
public class WebClient {

    //restante do código

    public void insere(String json) {

    }

}
```

Conforme vismo em aula, para essa requisição faremos praticamente o mesmo código contido no método `post` da classe `WebClient`:

```
public String post(String json) {
        try {
            URL url = new URL("https://www.caelum.com.br/mobile");
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestProperty("Content-type", "application/json");
            connection.setRequestProperty("Accept", "application/json");

            connection.setDoOutput(true);

            PrintStream output = new PrintStream(connection.getOutputStream());
            output.println(json);

            connection.connect();

            Scanner scanner = new Scanner(connection.getInputStream());
            String resposta = scanner.next();
            return resposta;
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
```

Refatorando código do `WebClient`

Apenas variando o valor da URL. Em outras palavras, refatorar o código para reutilizarmos nesse novo método que criamos. Portanto, extraia a variável URL para fora do `try`:

```
public String post(String json) {
    String endereco = "https://www.caelum.com.br/mobile";
    try {
        URL url = new URL(endereco);
    //restante do código
}
```

Observe que estamos utilizando agora a variável `endereco` para indicar a URL que será utilizada. Agora, selecione todo o código do `try` até o `return null`, e então, clique com o botão direito > **Refactor** > **Extract** > **Method**. Em seguida dê um nome para o método, nesse caso eu vou deixar como `realizaRequisicao`, pode adicionar o nome que desejar, desde que faça sentido para esse código! Caso esteja utilizando os atalhos do Eclipse, ao invés de fazer todo o procedimento mencionado, basta apenas usar o atalho **Alt + Shift + M**. Por fim, clique em **OK**. Então teremos o seguinte resultado:

```
public class WebClient {
    public String post(String json) {
        String endereco = "https://www.caelum.com.br/mobile";
        return realizaRequisicao(json, endereco);
    }

    @Nullable
    private String realizaRequisicao(String json, String endereco) {
        //código complexo que realiza requisicao
    }

    public void insere(String json) {

    }
}
```

Pegando o endereço do servidor

Agora precisamos chamar o método `realizaRequisicao` enviando o endereço e o JSON. Porém, qual endereço iremos enviar? Como vimos na aula, não podemos enviar para o endereço **localhost**, pois esse endereço é o famoso *loopback*, ou seja, é o endereço que aponta para o dispositivo ou emulador! Em outras palavras, precisamos pegar o IP da nossa máquina, pois é ela que está rodando o servidor. Para pegar o endereço do IP precisamos realizar os seguintes passos:

- **Linux e Mac**: Abra o terminal e digite o comando `ifconfig`, e pegue o valor do campo **inet addr:**, no meu caso o IP é 192.168.83.206
- **Windows**: Abra o CMD e digite `ipconfig`, pegue o valor do campo **ipv4**. Nesse caso, se estiver numa rede cabeada, veja o endereço do adaptador de Ethernet, se for Wi-Fi pegue a informação do adaptador de rede sem fio.

Observações: Para que funcione o dispositivo deve estar na mesma rede! Em outras palavras, se for emulador, não se preocupe, ambos (emulador e máquina) estão na mesma rede, porém, se for um celular seu, caso ele esteja numa rede diferente da sua máquina, não funcionará! Portanto, se pretende usar o seu celular, tenha certeza de que ele esteja conectado na mesma rede do seu computador que está executando o servidor.

Adicionando endereço da API do Web Service

Agora que temos o endereço IP basta apenas adicionarmos o endereço do Web Service que está esperando a requisição que insere um aluno em sua base da dados. Baseado no IP da minha máquina, temos o seguinte endereço para realizar a inserção:

```
    public void insere(String json) {
        String endereco = "http://192.168.83.206:8080/api/aluno";
    }
```

Em outras palavras, estamos acessando a minha máquina (**192.168.83.206**) na porta 8080 e com o caminho `"/api/aluno"` que é justamente o endereço para inserir um aluno via requisição HTTP. Agora basta apenas chamar o método `realizaRequisicao()` enviando o `json` e `endereco` via parâmetro respectivamente. A única diferença é que você terá que colocar o seu IP ao invés do meu! Faça as modificações necessárias.