05

Autenticando acessos à API

Para autenticar as requisições à API, pediremos que o usuário insira o código dele nos cabeçalhos da requisição a fim de não sujar a URL. Para acessar os cabeçalhos, precisamos acessar a requisição a partir do contexto atual e assim pegar o cabeçalho que quisermos, no caso, **API-Token**.

```
package autenticadores;
public class AcessoDaApiAutenticado extends Authenticator {
    @Inject
    private UsuarioDAO usuarioDAO;
    @Override
    public String getUsername(Context context) {
        String codigo = context.request().getHeader("API-Token");
        Optional<Usuario> possivelUsuario = usuarioDAO.comToken(codigo);
        if (possivelUsuario.isPresent()) {
            return possivelUsuario.get().getNome();
        }
        return null;
    }
}
```

Vamos sobrescrever também o método `onUnauthorized()` para retornar uma mensagem de erro em *JSON* caso o usuário não esteja autenticado, ao invés de enviar uma página em *HTML*. Ao invés de retornar um corpo com status `ok` ou `badRequest`, vamos utilizar o status `unauthorized`, mais adequado para a situação.

```
@Override
public Result onUnauthorized(Context context) {
    Map<String,String> parametrosDoErro = new HashMap<>();
    parametrosDoErro.put("código", "401");
    parametrosDoErro.put("mensagem", "Não autorizado!");
    Map<String,Object> erros = new HashMap<>();
    erros.put("errors", parametrosDoErro);
    return unauthorized(Json.toJson(erros));
}
```

Finalmente podemos adicionar a anotação de autenticação no **ApiController**. Faremos isso direto sobre o controller, pois assim garantimos que **todos** os métodos requerem autenticação.

```
@Authenticated(AcessoDaApiAutenticado.class)
public class ApiController extends Controller { ... }
```