Enviando nossa mensagens

Com o projeto montada e a API rodando, vamos então ao que interessa :)

Nosso primeiro passo, será enviar mensagens para a API, já que temos um campo de texto e um botão. Queremos enviar a mensagem que está escrita no EditText à API ao clicarmos no botão.

Para isso, não há mistério. Primeiro temos que buscar uma referência do botão e do campo de texto, logo vamos adicionar um id neles, lá no **activity\_main.xml**:

<EditText  
 android:id="@+id/et\_texto"  
 android:layout\_weight="2"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content" />  
  
<Button  
 android:id="@+id/btn\_enviar"  
 android:layout\_weight="1"  
 android:text="Enviar"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content" />

Agora, na *activity*, conseguimos pegar a referência do botão e do campo de texto, e vamos colocá-los como atributos da classe:

public class MainActivity extends AppCompatActivity {  
  
 private int idDoCliente = 1;  
 private Button button;  
 private EditText editText;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 ListView listaDeMensagens = (ListView) findViewById(R.id.lv\_mensagem);  
  
 List<Mensagem> mensagens = Arrays.asList(new Mensagem(1, "olá alunos de android"), new Mensagem(2, "oi"));  
  
 MensagemAdapter adapter = new MensagemAdapter(idDoCliente, mensagens, this);  
  
 listaDeMensagens.setAdapter(adapter);  
  
 button = (Button) findViewById(R.id.btn\_enviar);  
  
 editText = (EditText) findViewById(R.id.et\_texto);  
 }

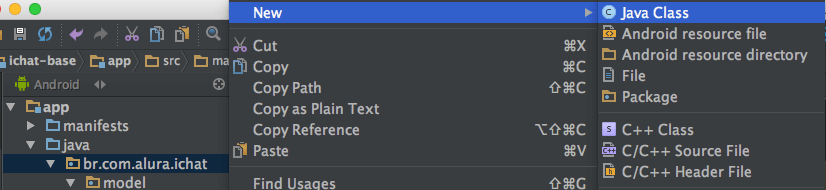
E vamos precisar associar um **listener** ao botão:

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 ListView listaDeMensagens = (ListView) findViewById(R.id.lv\_mensagem);  
  
 List<Mensagem> mensagens = Arrays.asList(new Mensagem(1, "olá alunos de android"), new Mensagem(2, "oi"));  
  
 MensagemAdapter adapter = new MensagemAdapter(idDoCliente, mensagens, this);  
  
 listaDeMensagens.setAdapter(adapter);  
  
 button = (Button) findViewById(R.id.btn\_enviar);  
  
 editText = (EditText) findViewById(R.id.et\_texto);  
  
 button.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 // aqui enviaremos a mensagem !!  
 }  
 });  
}

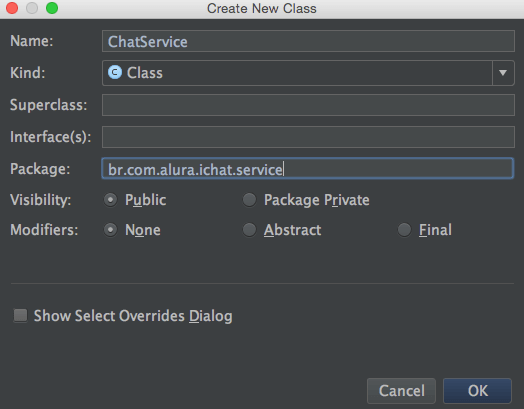
Depois precisamos pegar a mensagem do EditText e enviar. Algo como:

// MainActivity.java  
  
button.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 new ChatService().enviar(new Mensagem(idDoCliente, editText.getText().toString());  
 }  
});

Mas ainda não temos a classe ChatService no nosso projeto, precisaremos criá-la. Portanto, clique no pacote do projeto com o botão direito e pressione CTRL + N ou Command + N no Mac e selecione **New** > **Java Class**:



Ao abrir a janela, vamos dar o nome à classe de ChatService e colocá-la no pacote br.com.caelum.ichat.service:



Vamos criar um método chamado de enviar(String mensagem) na classe, por enquanto sem implementação:

public class ChatService {  
  
 public void enviar(Mensagem mensagem) {  
  
 }  
}

Vamos fazer uma requisição POST usando o pacote java.io que já conhecemos. Se isso for uma novidade, recomendo que você assista [*essa aula*](https://cursos.alura.com.br/course/android-studio-ii-integracoes-e-recursos/section/5/task/3) do treinamento **Android II: Integração com apps e recursos do device**.

public class ChatService {  
  
 public void enviar(Mensagem mensagem) {  
  
 String texto = mensagem.getTexto();  
  
 try {  
  
 // Altere para o seu IP  
 URL url = new URL("http://192.168.0.208:8080/polling");  
  
 HttpURLConnection httpConnection = (HttpURLConnection) url.openConnection();  
  
 httpConnection.setRequestMethod("POST");  
  
 httpConnection.setRequestProperty("Content-type", "application/json");  
  
 JSONStringer json = new JSONStringer()  
 .object()  
 .key("text")  
 .value(texto)  
 .key("id")  
 .value(mensagem.getId())  
 .endObject();  
  
 OutputStream saida = httpConnection.getOutputStream();  
 PrintStream ps = new PrintStream(saida);  
  
 ps.println(json.toString());  
  
 httpConnection.connect();  
 httpConnection.getInputStream();  
  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
  
 }  
  
}

Vamos agora rodar o aplicativo? Rode, digite um texto qualquer e pressione o botão **Enviar**. Será que conseguimos enviar a mensagem?

Parece que não, isso porque precisamos pedir permissão para usar a internet. Então, no arquivo **app/manifests/AndroidManifest.xml**, adicionamos a seguinte linha dentro da tag **<manifest>**:

<uses-permission android:name="android.permission.INTERNET" />

Rodamos novamente, enviamos uma mensagem e recebemos uma exceção do tipo android.os.NetworkOnMainThreadException. Isso significa que não podemos disparar requisições web na Thread principal do Android que é dedicada a tarefas relacionadas com a tela. Portanto, precisamos criar uma nova Thread:

public class ChatService {  
  
 public void enviar(Mensagem mensagem) {  
  
 new Thread(new Runnable() {  
  
 @Override  
 public void run() {  
  
 String texto = mensagem.getTexto();  
  
 try {  
  
 // Altere para o seu IP  
 URL url = new URL("http://192.168.0.208:8080/polling");  
  
 HttpURLConnection httpConnection = (HttpURLConnection) url.openConnection();  
  
 httpConnection.setRequestMethod("POST");  
  
 httpConnection.setRequestProperty("Content-type", "application/json");  
  
 JSONStringer json = new JSONStringer()  
 .object()  
 .key("text")  
 .value(texto)  
 .key("id")  
 .value(mensagem.getId())  
 .endObject();  
  
 OutputStream saida = httpConnection.getOutputStream();  
 PrintStream ps = new PrintStream(saida);  
  
 ps.println(json.toString());  
  
 httpConnection.connect();  
 httpConnection.getInputStream();  
  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
 }  
 }).start();  
 }  
}

**OBSERVAÇÃO:** Como estamos usando uma classe anônima, será necessário transformar o parâmetro mensagem do método em **final**, caso contrário haverá um erro de compilação. Para saber mais detalhes, leia o **Para saber mais** deste capítulo.

public void enviar(final Mensagem mensagem) {  
 // ...  
}

Agora enviamos uma mensagem, abrimos o endereço da API no navegador e conseguimos ver que a mensagem está sendo exibida!