



DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS

PROF^a. M.Sc. JULIANA H Q BENACCHIO



Tarefas Assíncronas

- O Android tem a classe **AsyncTask**, que facilita a comunicação de uma **thread** arbitrária com a **UI thread**.
- Para criar uma tarefa é preciso criar uma subclasse de **AsyncTask** e informar os três argumentos das classe genéricas (*Generics*) do Java **<Params, Progress, Result>**.



AsyncTask



```
public class MyTask extends AsyncTask<Integer, Integer, Integer> {  
  
    protected Integer doInBackground(Integer... args) {  
    }  
  
    protected void onPreExecute() {  
    }  
  
    protected void onProgressUpdate(Integer... values) {  
    }  
  
    protected void onPostExecute(Integer result) {  
    }  
}  
  
MyTask task = new MyTask()  
task.execute(100);
```

Tipo genéricos da AsyncTask

AsyncTask<Params, Progress, Result>

- **Params**: São argumentos que podemos passar ao método **execute(params...)** para executar o **AsyncTask**.
- **Progress**: Pode ser utilizado para receber um valor inteiro, que representa o progresso da execução, e em conjunto com um barra de progresso, para notificar o usuário.



Tipo genéricos da AsyncTask

- **Result:** É o mesmo objeto que retorna do método **doInBackground()** e é passado como parâmetro para o método **onPostExecute()**.



Métodos da AsyncTask

- Os métodos executam em **threads** diferentes.
- UI Thread
 - **onPreExecute()**
 - **onProgressUpdate()**
 - **onPostExecute()**
- Background Thread
 - **doInBackground()**



Métodos da AsyncTask

- **onPreExecute()** : Método executado antes da thread iniciar.
- **doInBackground()** : Método executado em background por uma thread, que deve conter todo o processamento pesado. Ele pode retornar um objeto qualquer, o qual será passado como parâmetro para o método **onPostExecute()**.



Métodos da AsyncTask

- **onProgressUpdate()**: Método que recebe geralmente um inteiro como parâmetro para informar a quantidade do progresso. O progresso deve ser informado dentro do método **doInBackground()** através da chamada do método **publishProgress(int)**.
- **onPostExecute()**: Método onde podemos atualizar a view com o resultado. Ele é chamado utilizando um Handler internamente.



- Regras para que as **AsyncTask** funcionem
 - O método **doInBackground()** não pode interagir com elementos da interface gráfica
 - A instância de **AsyncTask** deve ser criada na **UI Thread**
 - O método **execute()** deve ser invocado na **UI Thread**
 - Não se deve executar diretamente os métodos herdados de **AsyncTask**
 - A tarefa pode ser executada apenas uma vez



- O método **runOnUiThread()** usa um **handler** internamente
 - Este método é apenas um atalho para o uso de **handlers**
- Tarefas assíncronas (**AsyncTask**) gerenciam o uso de múltiplas **threads**
 - **AsyncTask** é apenas um atalho para a criação de **threads** manualmente



- Resumindo: Você deve criar um **AsyncTask** e utilizar o método **doInBackground()** para fazer o processamento, o qual é executado automaticamente em uma **thread**. Quando terminar, utilize o método **onPostExecute()** para atualizar as **views**. A classe **AsyncTask** elimina a necessidade de criar uma **thread** e utilizar um **handler**, pois ela já faz esse trabalho para você.

