# Задачка

- Направете приложение, което показва списък от коли със снимка и текст.
- Когато кликнете на елемент от списъка приложението отваря нов екран с детайлна информация за колата


# Android

# Background work

# Съдържание

- AsyncTask
- Broadcasts
- IntentService
- Service

# asynctask

# Защо съществува?

- Използва се за бекграунд операции, но и когато трябва те да влияят на потребителския интерфейс
- UI на апп-а се променя само от основната /наричана още UI/ нишка, за това са нужни специални механизми за да може той да се ъпдейтва от бекграунд процеси
- Да се използва основно за локални, кратки бекграунд процеси
-

# А защо не просто Thread?

```
public void onClick(View v) {
    new Thread(new Runnable() {
        public void run() {
            Bitmap b = loadImageFromNetwork("http://example.com/image.png");
            mImageView.setImageBitmap(b);
        }
    }).start();
}
```

- Не
- Може да се оправи с `Activity.runOnUiThread(Runnable)`
- Сложният код става по-сложен така

# Идеалният вариант

```
new DownloadImageTask().execute("http://example.com/image.png");
```

- Трябва да се наследи AsyncTask<String, Void, Bitmap>
  - Първия тип показва типа на входните данни
  - Втория тип - на прогреса
  - Третия на резултата
- Bitmap doInBackground(String... urls)
  - Единствения метод от класа, който е на отделна нишка
- void onPostExecute(Bitmap result)

# broadcast

# BroadcastReceiver

- Клас, който получава broadcasts
- Broadcasts са съобщения, които, когато са пратени, достигат до всички приложения
- Broadcasts се пращат когато
  - Се вкл/изкл wi-fi, bluetooth, друго
  - Когато батерията стане критична, когато се включи зарядно
  - Когато телефона се е рестартирал
  - При всяко по-важно събитие свързано с телефона
  - Когато вие самите изпратите ваш собствен broadcast
- Публичните Broadcasts могат да се прихванат от всички приложения
- Има и локални, които се изпращат само до елементите на вашето приложение

# Broadcasts

- За да получавате Broadcasts, трябва да си регистрирате receiver
  - `registerReceiver(BroadcastReceiver, IntentFilter)`
  - В intentFilter е дефиниран типа интенти за който слушате
- За да изпратите Broadcast трябва да използвате
  - ```
    Intent intent = new Intent();
          intent.setAction("com.tutorialspoint.CUSTOM_INTENT");
          sendBroadcast(intent);
    ```
  -
    ```

# intent service

# Прилики и разлики

- Wrapper на Service
- Живота й зависи от живота на приложението, за разлика от Service
- По-самостоятелна от AsyncTask

```
public class RSSPullService extends IntentService {
    @Override
    protected void onHandleIntent(Intent workIntent) {
        // Gets data from the incoming Intent
        String dataString = workIntent.getDataString();
        ...
        // Do work here, based on the contents of dataString
        ...
    }
}
```

# Извикване

- Извиква се чрез експлицитен интент и метода startService()
- Може да й се подаде информация като тя се запише в
- Връща информация на другите компоненти чрез BroadcastReceiver

```
Intent localIntent =
            new Intent(Constants.BROADCAST_ACTION)
            // Puts the status into the Intent
            .putExtra(Constants.EXTENDED_DATA_STATUS, status);
    // Broadcasts the Intent to receivers in this app.
    LocalBroadcastManager.getInstance(this).sendBroadcast(localIntent);
```

# Получаване на данните

```
// The filter's action is BROADCAST_ACTION
        IntentFilter mStatusIntentFilter = new IntentFilter(
                Constants.BROADCAST_ACTION);
// Instantiates a new DownloadStateReceiver
        DownloadStateReceiver mDownloadStateReceiver =
                new DownloadStateReceiver();
// Registers the DownloadStateReceiver and its intent filters
        LocalBroadcastManager.getInstance(this).registerReceiver(
                mDownloadStateReceiver,
                mStatusIntentFilter);
```

# service

# Как работи

- Ако сервиз е стартиран чрез метода `startService()` от активити или фрагмент, но живота му не зависи от живота на компонента, който го е стартирал
- Стартиран по този начин сервиз обикновено има определена цел и щом я изпълни трябва сам да се спре
- Сервиз може да се стартира и чрез `bindService()`. Много компоненти могат да се закачат за един и същ сервиз, чак когато всички са се разкачили от сервиза, той умира
- `onStartCommand()` - извиква се когато сервиза е стартиран
- `onBind()` - вика се когато друг компонент се закачи за сервиза

# домашно

# Задача

Създайте приложение, което стартира сервиз, който през 10 секунди записва местоположението на потребителя в SharedPreferences

В SharedPreferences се пише с:

- PreferenceManager.*getDefaultSharedPreferences*(context).edit().putString(KEY, locations).commit();

Чете се с:

- String locations = PreferenceManager.*getDefaultSharedPreferences*(context).getString(KEY);

При отваряне на приложението, всичката записана информация да се показва на екрана.

# въпроси

# ресурси

# Съдържание

- http://developer.android.com/guide/components/processes-and-threads.html
- http://developer.android.com/training/run-background-service/create-service.html
- Показване на картинки в лист
  - http://developer.android.com/training/displaying-bitmaps/index.html