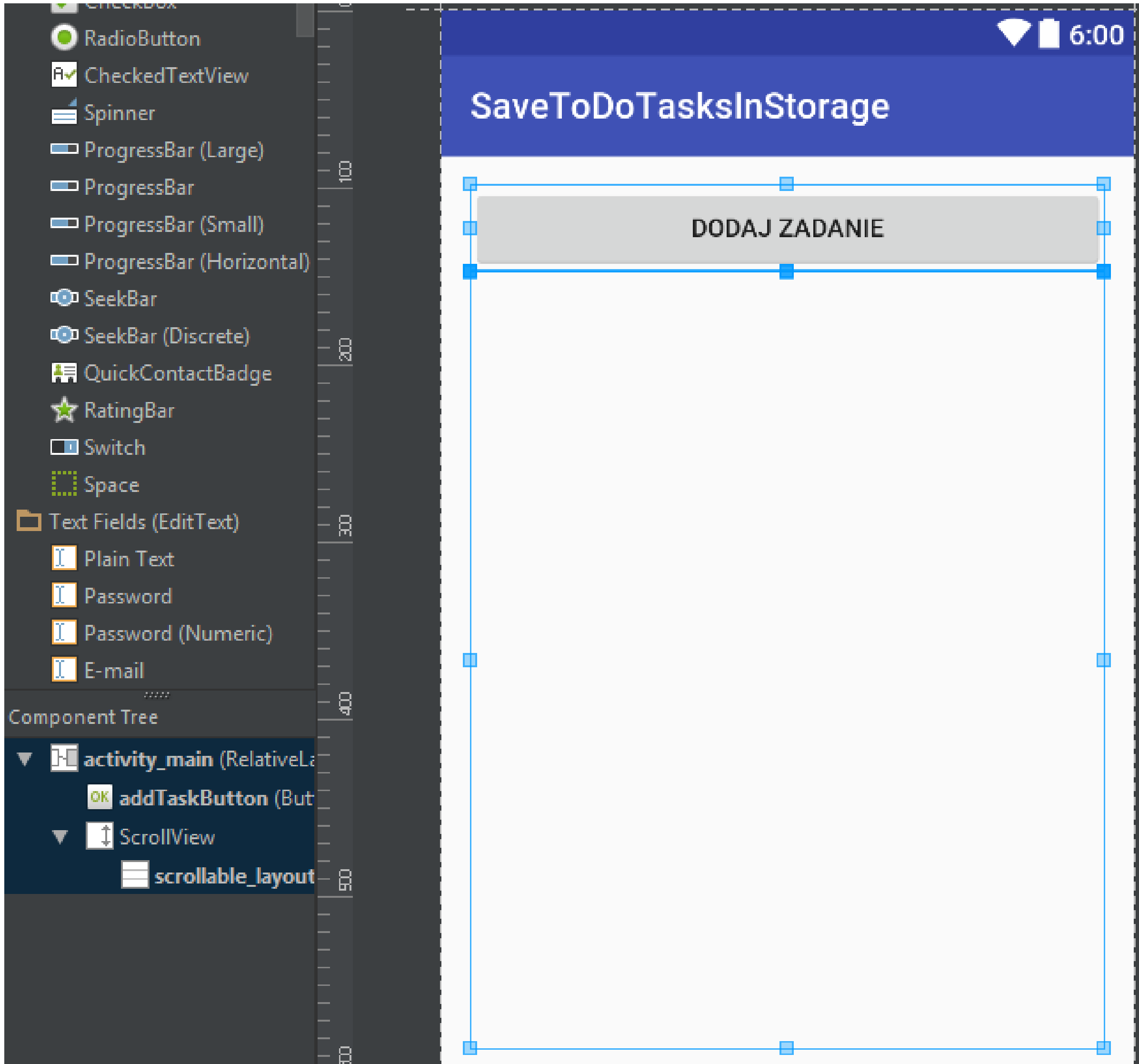
Java Android

SaveToDoTasksStorage

SaveToDoTasksStorage

Create user interface looking like this:



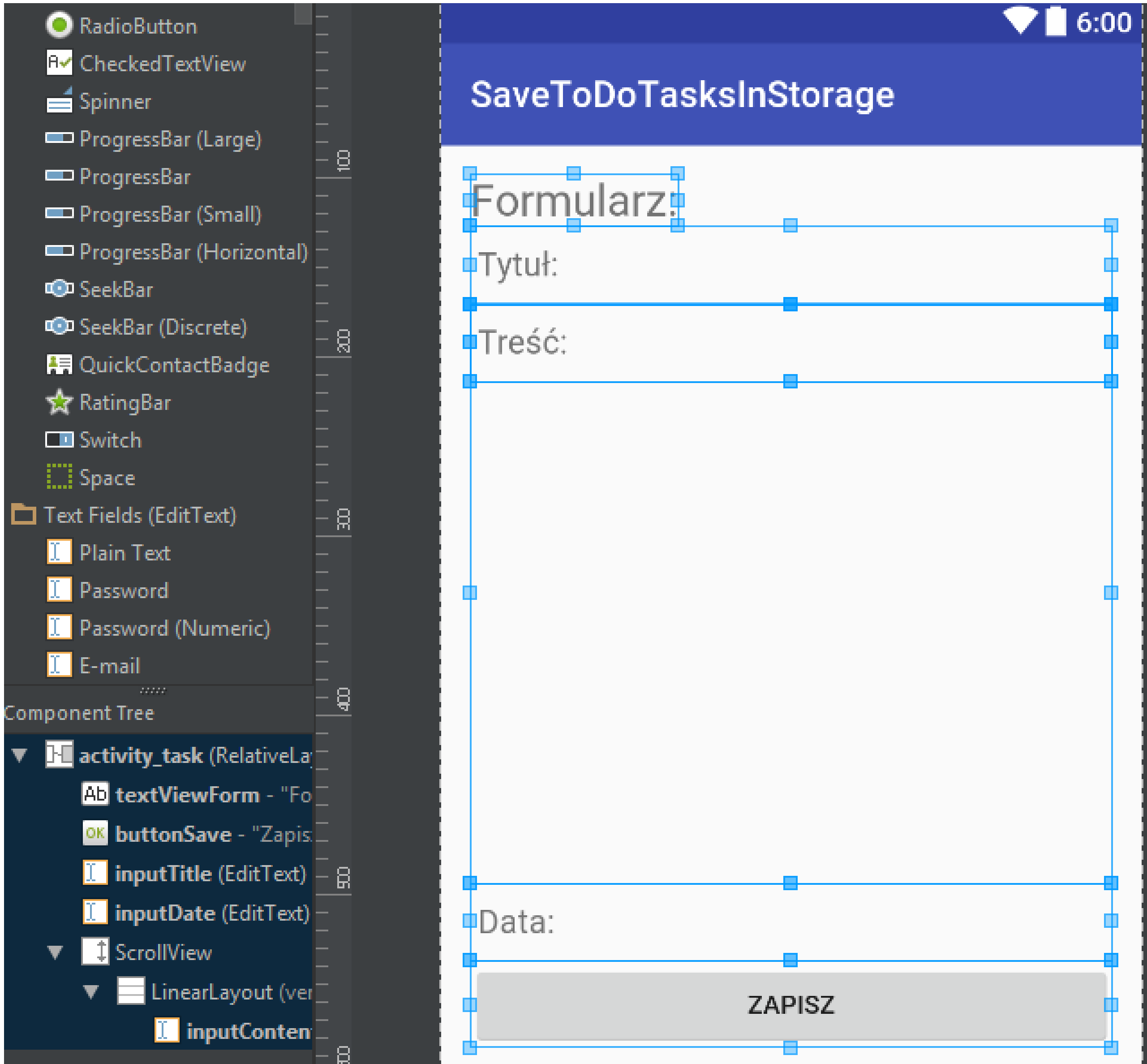
SaveToDoTasksStorage

Create user interface looking like this:

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.amen.savetodotasksinstorage.MainActivity">

    <Button
        android:text="Dodaj zadanie"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/addTaskButton" />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/addTaskButton"
        android:layout_alignParentStart="true"
        android:background="@drawable/back">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:id="@+id/scrollable_layout" />
    </ScrollView>
</RelativeLayout>
```

SaveToDoTasksStorage

Second looking like this:



SaveToDoTasksStorage

Second looking like this:

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_task"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"

tools:context="com.example.amen.savetodotasksinstorage.TaskActivity">

    <TextView
        android:text="Formularz:"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true"
        android:id="@+id/textViewForm"
        android:textSize="24sp" />

    <Button
        android:text="Zapisz"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/buttonSave" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:layout_below="@+id/textViewForm"
        android:layout_alignParentStart="true"
        android:id="@+id/inputTitle"
        android:layout_alignParentEnd="true"
        android:hint="Tytuł:" />
```

```xml
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="datetime"
        android:layout_above="@+id/buttonSave"
        android:layout_alignParentStart="true"
        android:id="@+id/inputDate"
        android:layout_alignParentEnd="true"
        android:hint="Data:"
        android:enabled="false"/>

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/inputTitle"
        android:layout_alignParentStart="true"
        android:layout_above="@+id/inputDate">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical" >

            <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="textMultiLine"
                android:ems="10"
                android:id="@+id/inputContent"
                android:hint="Treść:" />
        </LinearLayout>
    </ScrollView>
</RelativeLayout>
```

SaveToDoTasksStorage

Using ToDoTask description, we can define it’s class:

```java
public class ToDoTask {

    private Date data;
    private String tytul;
    private String tresc;

    public ToDoTask(String pTytul, String pTresc, Date pDate) {
        data = pDate;
        tytul = pTytul;
        tresc = pTresc;
    }

    public String getTitle() {
        return tytul;
    }

    public String getContent() {
        return tresc;
    }

    public String getDateString() {
        return new SimpleDateFormat("yyyy-MM-dd HH:mm").format(data);
    }
}
```

This class definition comes from desctiption. Each Task has date, title, and content. These are fields. Additionally we added their getters and basic constructor.

SaveToDoTasksStorage

Let’s start from MainActivity. Bind variables and define button’s onClickListener:

```java
private Button btn;
private LinearLayout layout;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    layout = (LinearLayout) findViewById(R.id.scrollable_layout);

    btn = (Button) findViewById(R.id.addTaskButton);
    btn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent i = new Intent(MainActivity.this, TaskActivity.class);
            startActivity(i);
        }
    });
}
```

This should be clear. After data binding we defined buttons onClick Listener to start second activity using startActivity method.

SaveToDoTasksStorage

Ok, that's all for now for the MainActivity. We'll come back to it later. Now TaskActivity:

```
private EditText content, title, date;
private Button saveBtn;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_task);

    content = (EditText) findViewById(R.id.inputContent);
    date = (EditText) findViewById(R.id.inputDate);
    title = (EditText) findViewById(R.id.inputTitle);

    saveBtn = (Button) findViewById(R.id.buttonSave);
    saveBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // create and add task to file
            finish();
        }
    });
}
```

After controls binding saveBtn action was defined to finish activity, but now we need to define task creation and saving it to files. Also in the onCreate method, we need to define setting date EditText value to current date:

```
date.setText(new SimpleDateFormat("yyyy-MM-dd HH:mm").format(new Date()));
```

SaveToDoTasksStorage

Creating ToDoTask using controls:

```
private ToDoTask createToDoTask() {
    String titleString = title.getText().toString();
    String contentString = content.getText().toString();
    Date currentDate = new Date();
    try {
        currentDate = new SimpleDateFormat("yyyy-MM-dd HH:mm").parse(date.getText().toString());
    } catch (ParseException e) {
        Toast.makeText(getApplicationContext(), "Wrong date...", Toast.LENGTH_SHORT).show();
    }

    return new ToDoTask(titleString, contentString, currentDate);
}
```

Ok, now we have almost everything, but we need some FileManager.

SaveToDoTasksStorage

I will define singleton FileManager. It will be Class which will be responsible for saving and loading data from and to file.

```java
public class FileManager {
    private static final String FILE_NAME = "todotasks.txt";

    public static final FileManager instance = new FileManager();
    private final List<ToDoTask> list;

    private FileManager() {
        list = new LinkedList<>();
    }

    public List<ToDoTask> getList() {
        return list;
    }

    private void saveToFile(List<ToDoTask> listToSave) {

    }

    private List<ToDoTask> getFromFile() {

    }
}
```

SaveToDoTasksStorage

I will first implement saveToFile:

```java
private void saveToFile(Context context, List<ToDoTask> listToSave) {
    try {
        FileOutputStream ofs = context.openFileOutput(FILE_NAME, Context.MODE_PRIVATE);
        OutputStreamWriter writer = new OutputStreamWriter(ofs);
        PrintWriter printWriter = new PrintWriter(writer);

        for (ToDoTask task : listToSave) {
            printWriter.println(task.toSerializedString());
        }

        printWriter.close();
    } catch (FileNotFoundException fnfe) {
        Toast.makeText(context, "File not found.", Toast.LENGTH_SHORT).show();
    } catch (IOException ioe) {
        Toast.makeText(context, "File read error.", Toast.LENGTH_SHORT).show();
    }
}
```

Because i need context when opening internal storage, i had to pass it as a parameter to this method. After that i used OpenStreamWriter which was created from FileOutputStream, and from that i created PrintWriter which allows using additional methods on streams, such as println – which prints line of string and appends endline sign. For this method i also created toSerializedString method which will be defined on the next slide.

It's important to save the stream after it is not used.

SaveToDoTasksStorage

toSerializedString:

```
public String toSerializedString() {
    return tytul + SEPARATOR + new SimpleDateFormat("yyyy-MM-dd HH:mm").format(data) + SEPARATOR + tresc;
}
```

I’ve decided my file will look like this:
-Each line will be single object of ToDoTask type,
-Each line will be seperated by SEPARATOR which is constant defined:

```
private static final String SEPARATOR = ";";
```

-Each line consists of three variables separated by previously mentioned SEPARATOR, where first parameter is title, second is date, third is content.

Can anyone guess why content is last?
Does it have any meaning?

SaveToDoTasksStorage

Let’s do the getFrom file:

```java
private List<ToDoTask> getFromFile(Context context) {
    List<ToDoTask> returnList = new LinkedList<>();

    try {
        FileInputStream inputStreamReader = context.openFileInput(FILE_NAME);
        InputStreamReader streamReader = new InputStreamReader(inputStreamReader);
        BufferedReader reader = new BufferedReader(streamReader);

        String line = reader.readLine();

        while (line != null && !line.isEmpty()) {
            returnList.add(new ToDoTask(line));

            line = reader.readLine();
        }

        reader.close();
    } catch (FileNotFoundException fnfe) {
        Toast.makeText(context, "File not found.", Toast.LENGTH_SHORT).show();
    } catch (IOException ioe) {
        Toast.makeText(context, "Read exception.", Toast.LENGTH_SHORT).show();
    }

    return returnList;
}
```

Same way signature changes to allow us using context. Using it we are opening File, and after that similar to output stream we are opening input stream from which we can create PrintWriter. This object allows us to read single line using readLine method. We are reading all lines in a loop. Now, we need to create additional constructor in the ToDoTask, which will parse this line:

SaveToDoTasksStorage

Parsing line in ToDoTask in a constructor:

```java
public ToDoTask(String fromLine) {
    String[] splits = fromLine.split(SEPARATOR);

    tytul = splits[0];
    try {
        data = new SimpleDateFormat("yyyy-MM-dd HH:mm").parse(splits[1]);
    } catch (ParseException e) {
        e.printStackTrace();
    }
    tresc = splits[2];
}
```

Parsing previously created serialized line.

SaveToDoTasksStorage

Now we want to have file memory cached in the program, so we used List of ToDoTask in the FileManager. We don’t want users to save and load their own lists, we only want to be one list and we want it to be owned by FileManager, so we need to define only simple methods for save and load:

```java
// ładowanie do listy (która jest w pamięci aplikacji) wszystkich elementów z pliku
public void load(Context context){
    list.addAll(getFromFile(context));
}

// zapisywanie listy do pliku.
public void save(Context context){
    saveToFile(context, list);
}
```

SaveToDoTasksStorage

Now we want to have file memory cached in the program, so we used List of ToDoTask in the FileManager. We don’t want users to save and load their own lists, we only want to be one list and we want it to be owned by FileManager, so we need to define only simple methods for save and load:

```
// ładowanie do listy (która jest w pamięci aplikacji) wszystkich elementów z pliku
public void load(Context context){
    list.addAll(getFromFile(context));
}

// zapisywanie listy do pliku.
public void save(Context context){
    saveToFile(context, list);
}
```

We want loading to be invoked only once in the appliction, so it should be done in the MainActivity:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    //...

    FileManager.instance.load(getApplicationContext());
}
```

SaveToDoTasksStorage

But we want to save file content whenever it changes. And because we are assuming that it will change only whenever some file is added, than it’s implementation should be added in the btnSave onClick method:

```
saveBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ToDoTask task = createToDoTask();

        // add element to list in the FileManager
        FileManager.instance.getList().add(task);

        // save changes in the file by calling save on FileManager
        FileManager.instance.save(getApplicationContext());

        // finish activity
        finish();
    }
});
```

SaveToDoTasksStorage

Now let’s get back to the MainActivity where we left without calling onResume – in which our activity should repaint all buttons to show all saved in file tasks.

```java
@Override
protected void onResume() {
    super.onResume();

    layout.removeAllViews();
    List<ToDoTask> tasks = FileManager.instance.getList();
    LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(layout.getLayoutParams());
    params.topMargin = 15;

    for (final ToDoTask task : tasks) {
        Button btn = new Button(this);
        btn.setText(task.getTitle());
        btn.setBackgroundColor(Color.RED);
        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(MainActivity.this, TaskActivity.class);
                i.putExtra("Task", task.toSerializedString());
                startActivity(i);
            }
        });
        layout.addView(btn, params);
    }
}
```

This implementation first – removes all previous buttons from layout, than creaates button for each task in the FileManager and each button is added to the layout with different onClickListener. Whenever button is clicked it calls starting second activity again, but this time it is putting extra string under „Task” key.

SaveToDoTasksStorage

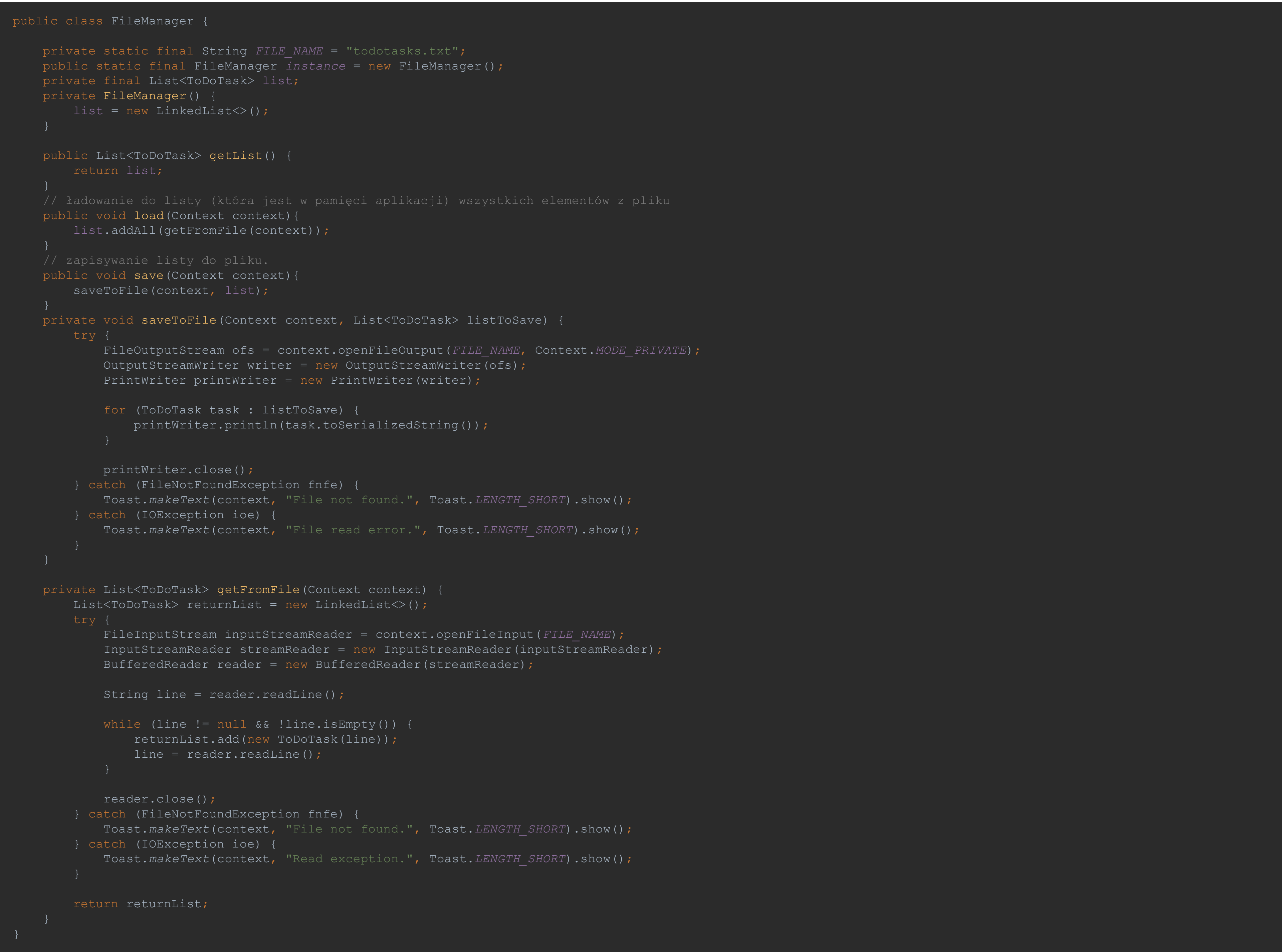
The last thing is handling intent data in the TaskActivity:

```
if (getIntent().hasExtra("Task")) {
    ToDoTask toSet = new ToDoTask(getIntent().getStringExtra("Task"));
    content.setEnabled(false);
    title.setEnabled(false);

    title.setText(toSet.getTitle());
    content.setText(toSet.getContent());
    date.setText(toSet.getDateString());
}
```

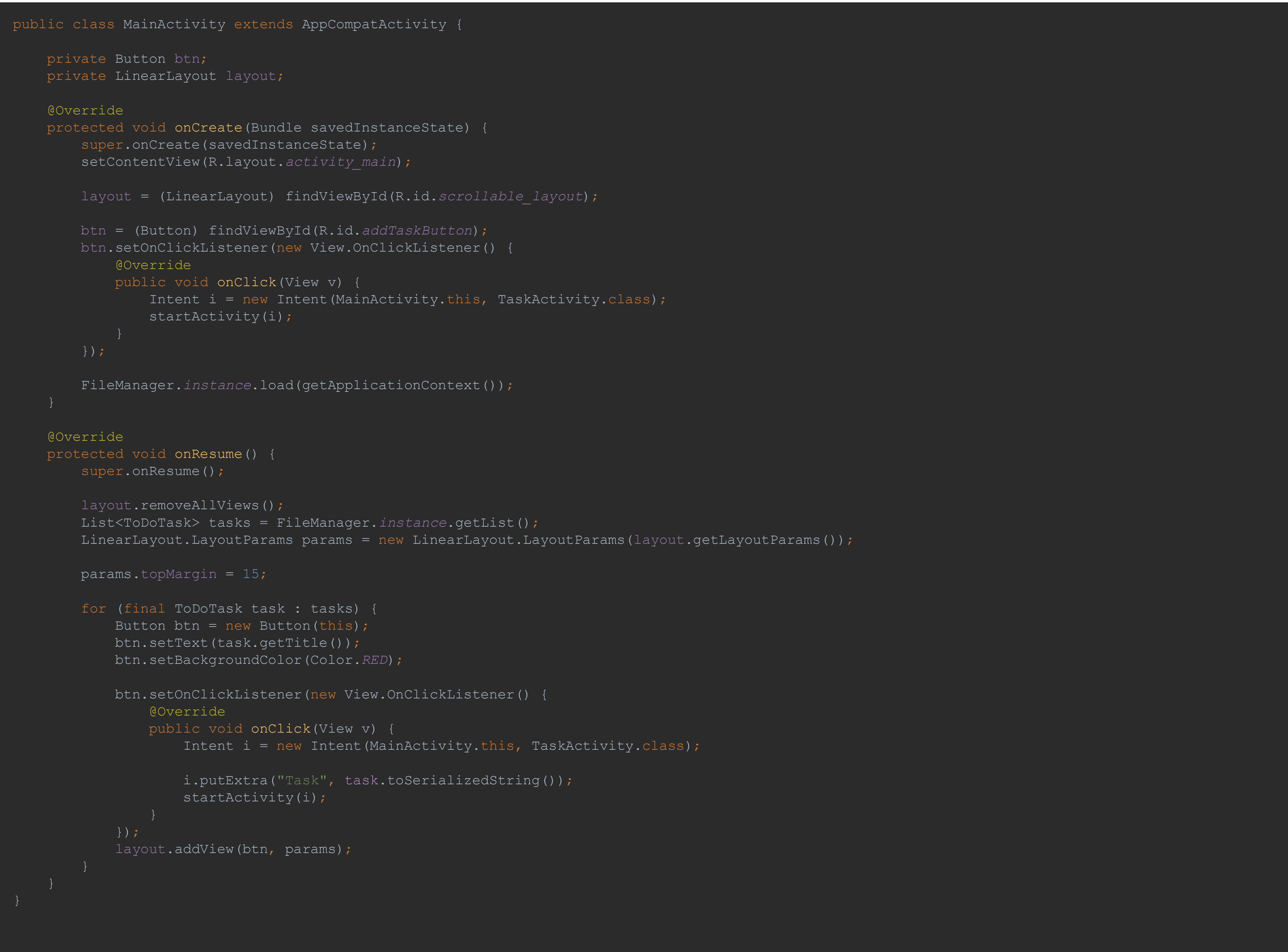
AND disallow user to add the same ToDoTask by adding intent check in the saveBtn OnClickListener:

```
saveBtn = (Button) findViewById(R.id.buttonSave);
saveBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (!getIntent().hasExtra("Task")) {
            ToDoTask task = createToDoTask();


            FileManager.instance.getList().add(task);
            FileManager.instance.save(getApplicationContext());
        }

        finish();
    }
});
```

```java
public class ToDoTask {

    private static final String SEPARATOR = ";";

    private Date data;
    private String tytul;
    private String tresc;

    public ToDoTask(String fromLine) {
        String[] splits = fromLine.split(SEPARATOR);

        tytul = splits[0];
        try {
            data = new SimpleDateFormat("yyyy-MM-dd HH:mm").parse(splits[1]);
        } catch (ParseException e) {
            e.printStackTrace();
        }
        tresc = splits[2];
    }

    public ToDoTask(String pTytul, String pTresc, Date pDate) {
        data = pDate;
        tytul = pTytul;
        tresc = pTresc;
    }

    public String toSerializedString() {
        return tytul + SEPARATOR + new SimpleDateFormat("yyyy-MM-dd HH:mm").format(data) + SEPARATOR + tresc;
    }

    public String getTitle() {
        return tytul;
    }

    public String getContent() {
        return tresc;
    }

    public String getDateString() {
        return new SimpleDateFormat("yyyy-MM-dd HH:mm").format(data);
    }
}
```



```java
public class FileManager {

    private static final String FILE_NAME = "todotasks.txt";
    public static final FileManager instance = new FileManager();
    private final List<ToDoTask> list;
    private FileManager() {
        list = new LinkedList<>();
    }

    public List<ToDoTask> getList() {
        return list;
    }
    // ładowanie do listy (która jest w pamięci aplikacji) wszystkich elementów z pliku
    public void load(Context context){
        list.addAll(getFromFile(context));
    }
    // zapisywanie listy do pliku.
    public void save(Context context){
        saveToFile(context, list);
    }
    private void saveToFile(Context context, List<ToDoTask> listToSave) {
        try {
            FileOutputStream ofs = context.openFileOutput(FILE_NAME, Context.MODE_PRIVATE);
            OutputStreamWriter writer = new OutputStreamWriter(ofs);
            PrintWriter printWriter = new PrintWriter(writer);

            for (ToDoTask task : listToSave) {
                printWriter.println(task.toSerializedString());
            }

            printWriter.close();
        } catch (FileNotFoundException fnfe) {
            Toast.makeText(context, "File not found.", Toast.LENGTH_SHORT).show();
        } catch (IOException ioe) {
            Toast.makeText(context, "File read error.", Toast.LENGTH_SHORT).show();
        }
    }

    private List<ToDoTask> getFromFile(Context context) {
        List<ToDoTask> returnList = new LinkedList<>();
        try {
            FileInputStream inputStreamReader = context.openFileInput(FILE_NAME);
            InputStreamReader streamReader = new InputStreamReader(inputStreamReader);
            BufferedReader reader = new BufferedReader(streamReader);

            String line = reader.readLine();

            while (line != null && !line.isEmpty()) {
                returnList.add(new ToDoTask(line));
                line = reader.readLine();
            }

            reader.close();
        } catch (FileNotFoundException fnfe) {
            Toast.makeText(context, "File not found.", Toast.LENGTH_SHORT).show();
        } catch (IOException ioe) {
            Toast.makeText(context, "Read exception.", Toast.LENGTH_SHORT).show();
        }

        return returnList;
    }
}
```



```
public class MainActivity extends AppCompatActivity {

    private Button btn;
    private LinearLayout layout;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        layout = (LinearLayout) findViewById(R.id.scrollable_layout);

        btn = (Button) findViewById(R.id.addTaskButton);
        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(MainActivity.this, TaskActivity.class);
                startActivity(i);
            }
        });

        FileManager.instance.load(getApplicationContext());
    }

    @Override
    protected void onResume() {
        super.onResume();

        layout.removeAllViews();
        List<ToDoTask> tasks = FileManager.instance.getList();
        LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(layout.getLayoutParams());

        params.topMargin = 15;

        for (final ToDoTask task : tasks) {
            Button btn = new Button(this);
            btn.setText(task.getTitle());
            btn.setBackgroundColor(Color.RED);

            btn.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Intent i = new Intent(MainActivity.this, TaskActivity.class);

                    i.putExtra("Task", task.toSerializedString());
                    startActivity(i);
                }
            });
            layout.addView(btn, params);
        }
    }
}
```

```java
public class TaskActivity extends AppCompatActivity {

    private EditText content, title, date;
    private Button saveBtn;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_task);

        content = (EditText) findViewById(R.id.inputContent);
        date = (EditText) findViewById(R.id.inputDate);
        title = (EditText) findViewById(R.id.inputTitle);

        saveBtn = (Button) findViewById(R.id.buttonSave);
        saveBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (!getIntent().hasExtra("Task")) {
                    ToDoTask task = createToDoTask();


                    FileManager.instance.getList().add(task);
                    FileManager.instance.save(getApplicationContext());
                }

                finish();
            }
        });

        date.setText(new SimpleDateFormat("yyyy-MM-dd HH:mm").format(new Date()));
        if (getIntent().hasExtra("Task")) {
            ToDoTask toSet = new ToDoTask(getIntent().getStringExtra("Task"));
            content.setEnabled(false);
            title.setEnabled(false);

            title.setText(toSet.getTitle());
            content.setText(toSet.getContent());
            date.setText(toSet.getDateString());
        }

        getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_HIDDEN);
    }

    private ToDoTask createToDoTask() {
        String titleString = title.getText().toString();
        String contentString = content.getText().toString();
        Date currentDate = new Date();
        try {
            currentDate = new SimpleDateFormat("yyyy-MM-dd HH:mm").parse(date.getText().toString());
        } catch (ParseException e) {
            Toast.makeText(getApplicationContext(), "Wrong date...", Toast.LENGTH_SHORT).show();
        }

        return new ToDoTask(titleString, contentString, currentDate);
    }
}
```

The end