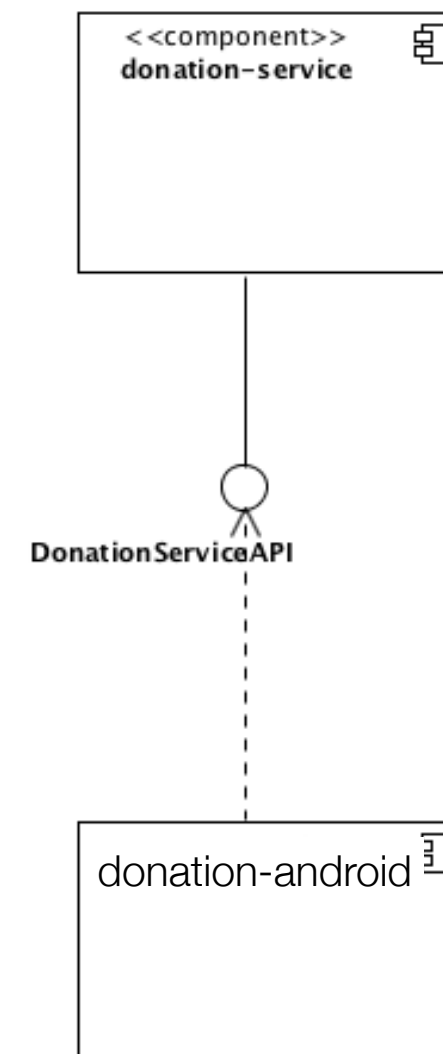


Mobile Application Development

Donation Android Client

- Use donation-service for
 - Login
 - Signup
 - Donate
 - List All Donations

GET	/api/donors
GET	/api/donors/{id}
POST	/api/donors
DELETE	/api/donors/{id}
DELETE	/api/donors
GET	/api/donations
DELETE	/api/donations
GET	/api/donors/{id}/donations
GET	/api/donors/{id}/donations/{donationId}
POST	/api/donors/{id}/donations
DELETE	/api/donors/{id}/donations/{donationId}



Android Project Configuration

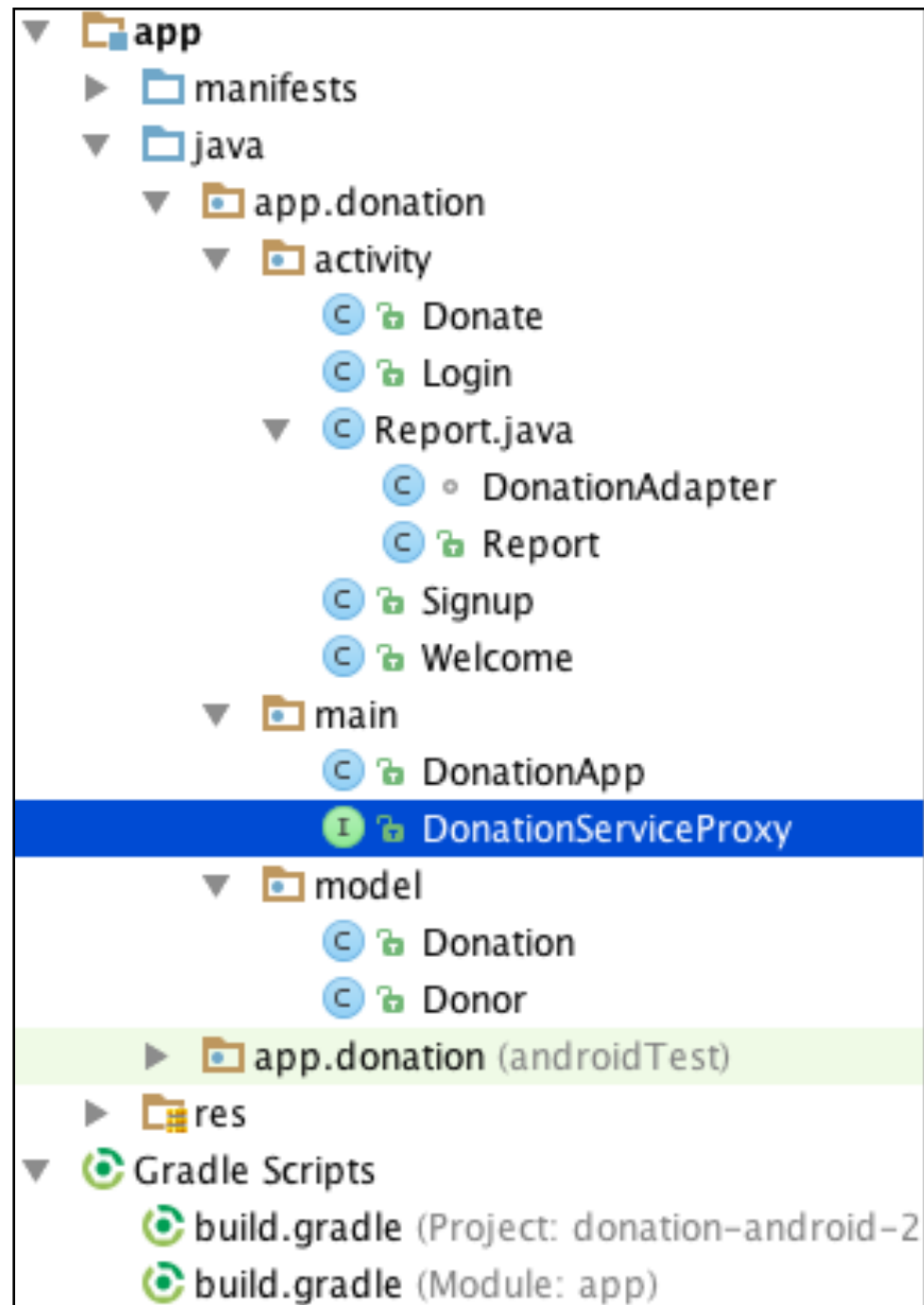
- Gradle - new libraries

```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    compile 'com.android.support:appcompat-v7:23.0.0'  
    compile 'com.google.code.gson:gson:2.4'  
    compile 'com.squareup.retrofit:retrofit:2.0.0-beta2'  
    compile 'com.squareup.retrofit:converter-gson:2.0.0-beta2'  
}
```

- AndroidManifest.xml
 - new 'permission' to access internet

```
<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="app.donation" >  
  
    <uses-permission android:name="android.permission.INTERNET"/>  
  
    <application  
        android:name=".main.DonationApp"  
        android:allowBackup="true"  
        android:icon="@mipmap/ic_launcher"
```

Android Project One New class - DonationServiceProxy



```
public interface DonationServiceProxy
{
    @GET("/api/donors")
    Call<List<Donor>> getAllDonors();

    @GET("/api/donors/{id}")
    Call<Donor> getDonor(@Path("id") Long id);

    @POST("/api/donors")
    Call<Donor> createDonor(@Body Donor donor);

    @DELETE("/api/donors/{id}")
    Call<Donor> deleteDonor(@Path("id") Long id);

    @DELETE("/api/donors")
    Call<String> deleteAllDonors();

    @GET("/api/donations")
    Call<List<Donation>> getAllDonations();

    @DELETE("/api/donations")
    Call<String> deleteAllDonations();

    @GET("/api/donors/{id}/donations")
    Call<List<Donation>> getDonations(@Path("id") Long id);

    @GET("/api/donors/{id}/donations/{donationId}")
    Call<Donation> getDonation(@Path("id") Long id, @Path("id") Long donationId);

    @POST("/api/donors/{id}/donations")
    Call<Donation> createDonation(@Path("id") Long id, @Body Donation donation);

    @DELETE("/api/donors/{id}/donatinos/{donationId}")
    Call<Donation> deleteDonation(@Path("id") Long id, @Path("id") Long donationId);
}
```

Android Models

- introduce ID field into both model objects

```
public class Donor
{
    public Long    id;
    public String  firstName;
    public String  lastName;
    public String  email;
    public String  password;

    public Donor(String firstName, String lastName, String email, String password)
    {
        this.firstName = firstName;
        this.lastName = lastName;
        this.email = email;
        this.password = password;
    }
}
```

```
public class Donation
{
    public Long    id;
    public int     amount;
    public String  method;

    public Donation (int amount, String method)
    {
        this.amount = amount;
        this.method = method;
    }
}
```

DonationApp - Attributes

```
public class DonationApp extends Application
{
    public String          service_url  = "http://10.0.2.2:9000";    // Standard Emulator IP Address

    public DonationServiceProxy donationService;
    public boolean         donationServiceAvailable = false;

    public Donor           currentUser;
    ...
}
```

- New fields:
 - 'service_url' string for remote service
 - donationService member for accessing the service
 - donationServiceAvailable flag if service offline
 - currentUser - the logged in user

DonationApp - onCreate

```
@Override
public void onCreate()
{
    super.onCreate();
    Gson gson = new GsonBuilder().create();

    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(service_url)
        .addConverterFactory(GsonConverterFactory.create(gson))
        .build();
    donationService = retrofit.create(DonationServiceProxy.class);
    Log.v("Donation", "Donation App Started");
}
```

- Create the proxy service 'donationService', with the appropriate Gson parsers

Signup - standalone version

```
public class Signup extends AppCompatActivity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_signup);
    }

    public void registerPressed (View view)
    {
        TextView firstName = (TextView) findViewById(R.id.firstName);
        TextView lastName = (TextView) findViewById(R.id.lastName);
        TextView email = (TextView) findViewById(R.id.Email);
        TextView password = (TextView) findViewById(R.id.Password);

        Donor user = new Donor(firstName.getText().toString(), lastName.getText().toString(),
                                email.getText().toString(), password.getText().toString());

        DonationApp app = (DonationApp) getApplication();
        app.newUser(user);

        startActivity (new Intent(this, Welcome.class));
    }
}
```

Signup

Sign up for the Donation App

Enter details below

First name

Last Name

Email

Password

REGISTER

Signup - Networked Version

```
public class Signup extends AppCompatActivity implements Callback<Donor>
{
    private DonationApp app;

    //.. as before

    public void registerPressed (View view)
    {
        //.. as before

        DonationApp app = (DonationApp) getApplication();
        Call<Donor> call = (Call<Donor>) app.donationService.createDonor(donor);
        call.enqueue(this);
    }

    @Override
    public void onResponse(Response<Donor> response, Retrofit retrofit)
    {
        app.donors.add(response.body());
        startActivity(new Intent(this, Welcome.class));
    }

    @Override
    public void onFailure(Throwable t)
    {
        app.donationServiceAvailable = false;
        Toast toast = Toast.makeText(this, "Donation Service Unavailable..",
                                     Toast.LENGTH_LONG);
        toast.show();
        startActivity (new Intent(this, Welcome.class));
    }
}
```

- Callback interface for response from service

- Service Call

- Callback Handlers

- for success

- for error

Welcome - standalone version

```
public class Welcome extends AppCompatActivity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_welcome);
    }

    public void loginPressed (View view)
    {
        startActivity(new Intent(this, Login.class));
    }

    public void signupPressed (View view)
    {
        startActivity (new Intent(this, Signup.class));
    }
}
```

Donation

LOGIN

SIGN UP

Welcome - networked version

```
public class Welcome extends AppCompatActivity implements Callback<List<Donor>>
{
    private DonationApp app;

    @Override
    public void onResume()
    {
        super.onResume();
        app.currentUser = null;
        Call<List<Donor>> call = (Call<List<Donor>>) app.donationService.getAllDonors();
        call.enqueue(this);
    }

    @Override
    public void onResponse(Response<List<Donor>> response, Retrofit retrofit)
    {
        serviceAvailableMessage();
        app.donors = response.body();
        app.donationServiceAvailable = true;
    }

    @Override
    public void onFailure(Throwable t)
    {
        app.donationServiceAvailable = false;
        serviceUnavailableMessage();
    }

    // ...
}
```

- Callback interface for response from service
- Service Call
- Callback Handlers
- for success
- for error

```
public class Welcome extends AppCompatActivity implements Callback<List<Donor>>
{
    //...

    public void loginPressed (View view)
    {
        if (app.donationServiceAvailable)
        {
            startActivity (new Intent(this, Login.class));
        }
        else
        {
            serviceUnavailableMessage();
        }
    }

    public void signupPressed (View view)
    {
        if (app.donationServiceAvailable)
        {
            startActivity (new Intent(this, Signup.class));
        }
        else
        {
            serviceUnavailableMessage();
        }
    }

    void serviceUnavailableMessage()
    {
        Toast toast = Toast.makeText(this, "Donation Service Unavailable. Try again later", Toast.LENGTH_LONG);
        toast.show();
    }

    void serviceAvailableMessage()
    {
        Toast toast = Toast.makeText(this, "Donation Contacted Successfully", Toast.LENGTH_LONG);
        toast.show();
    }
}
```

Donate - Standalone Version

```
public class Donate extends AppCompatActivity
{

    public void donateButtonPressed (View view)
    {
        String method = paymentMethod.getCheckedRadioButtonId() == R.id.PayPal ? "PayPal" : "Direct";
        int donatedAmount = amountPicker.getValue();
        if (donatedAmount == 0)
        {
            String text = amountText.getText().toString();
            if (!text.equals(""))
                donatedAmount = Integer.parseInt(text);
        }
        if (donatedAmount > 0)
        {
            app.newDonation(new Donation(donatedAmount, method));
            progressBar.setProgress(app.totalDonated);
            String totalDonatedStr = "$" + app.totalDonated;
            amountTotal.setText(totalDonatedStr);
        }
        amountText.setText("");
        amountPicker.setValue(0);
    }
}
```

Donation

Welcome Homer
Please give generously

☒ PayPal

☐ Direct

1000

0

1

Amount:

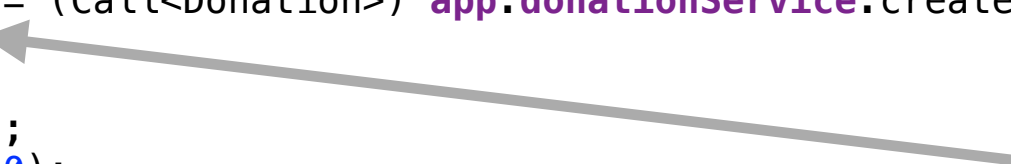
DONATE

Total so far: 0

Donate - Networked Version

```
public class Donate extends AppCompatActivity implements Callback<Donation>
{
    ...

    public void donateButtonPressed (View view)
    {
        String method = paymentMethod.getCheckedRadioButtonId() == R.id.PayPal ? "PayPal" : "Direct";
        int donatedAmount = amountPicker.getValue();
        if (donatedAmount == 0)
        {
            String text = amountText.getText().toString();
            if (!text.equals(""))
                donatedAmount = Integer.parseInt(text);
        }
        if (donatedAmount > 0)
        {
            Donation donation = new Donation(donatedAmount, method);
            Call<Donation> call = (Call<Donation>) app.donationService.createDonation(app.currentUser.id, donation);
            call.enqueue(this);
        }
        amountText.setText("");
        amountPicker.setValue(0);
    }
}
```



- Service Call

Donate - Networked Version

```
public class Donate extends AppCompatActivity implements Callback<Donation>
{
    ...

    @Override
    public void onResponse(Response<Donation> response, Retrofit retrofit)
    {
        Toast toast = Toast.makeText(this, "Donation Accepted", Toast.LENGTH_SHORT);
        toast.show();
        app.newDonation(response.body());
        progressBar.setProgress(app.totalDonated);
        String totalDonatedStr = "$" + app.totalDonated;
        amountTotal.setText(totalDonatedStr);
        amountText.setText("");
        amountPicker.setValue(0);
    }

    @Override
    public void onFailure(Throwable t)
    {
        Toast toast = Toast.makeText(this, "Error making donation", Toast.LENGTH_LONG);
        toast.show();
    }
}
```

- Service Response

Report - Standalone Version

```
public class Report extends AppCompatActivity
{
    private ListView    listView;
    private DonationApp app;

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_report);

        app = (DonationApp) getApplication();

        listView = (ListView) findViewById(R.id.reportList);
        DonationAdapter adapter = new DonationAdapter (this, app.donations);
        listView.setAdapter(adapter);
    }
    ...
}
```

Donation



Report

210	paypal
20	cash
330	cash
10	paypal
999	PayPal

Report - Networked Version

```
public class Report extends AppCompatActivity implements Callback<List<Donation>>
{
    private ListView        listView;
    private DonationApp      app;
    private DonationAdapter adapter;

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_report);

        app = (DonationApp) getApplication();

        listView = (ListView) findViewById(R.id.reportList);
        adapter = new DonationAdapter (this, app.donations);
        listView.setAdapter(adapter);

        Call<List<Donation>> call = (Call<List<Donation>>) app.donationService.getAllDonations();
        call.enqueue(this);
    }

    @Override
    public void onResponse(Response<List<Donation>> response, Retrofit retrofit)
    {
        adapter.donations = response.body();
        adapter.notifyDataSetChanged();
    }

    @Override
    public void onFailure(Throwable t)
    {
        Toast toast = Toast.makeText(this, "Error retrieving donations", Toast.LENGTH_LONG);
        toast.show();
    }

    //...
}
```