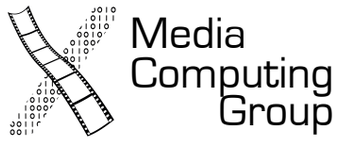


RWTH AACHEN UNIVERSITY

Mobile Application Development
L14: Miscellaneous

Jonathan Diehl (Informatik 10)
Hendrik Thüs (Informatik 9)

Agenda

- Accessing Contacts, Calendars, and Email
- Using the Camera
- User Preferences & Settings (iOS)
- Background Execution & Notifications

+ Review

Address Book

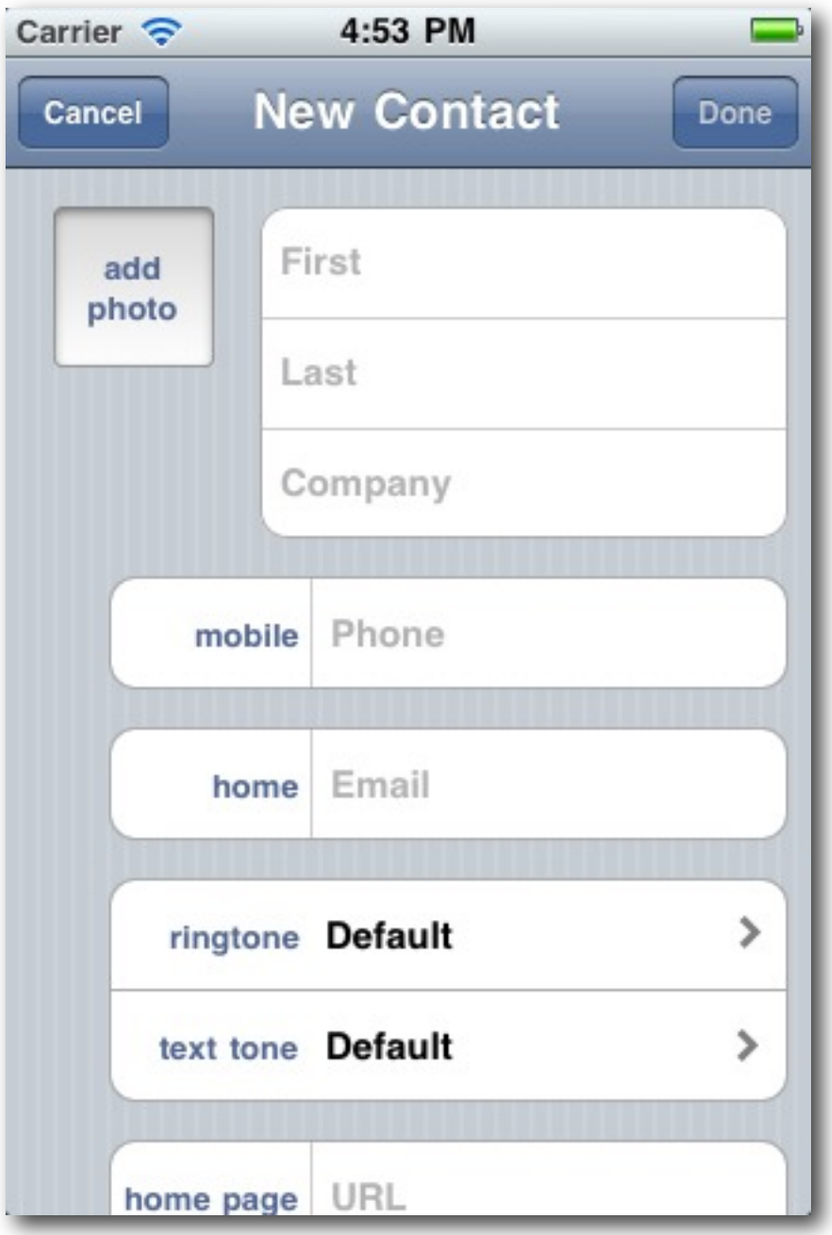
iOS: Address Book + UI

- Address Book
  - C Library
  - Programmatic access to the address book

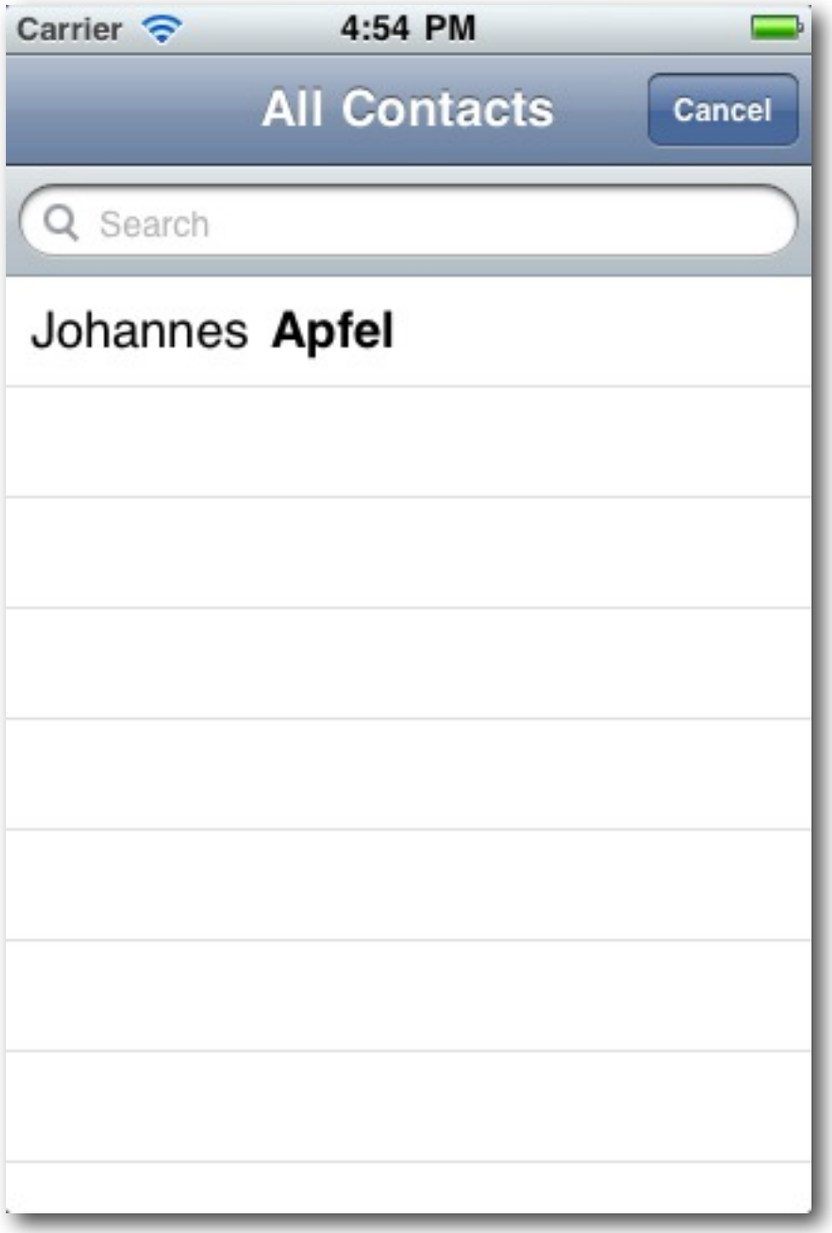
```
NSString *firstName = (NSString *)ABRecordCopyValue(person,
                                kABPersonFirstNameProperty);
...
[firstName release];
```

- Address Book UI
  - Objective-C Framework
  - View controllers for accessing the address book

Address Book UI



ABNewPersonViewController



ABPeoplePickerNavigationController

Android: Demo

Calendar

Android: Calendar via Intent

```
public void addNewEvent() {
Calendar cal = Calendar.getInstance();

Intent intent = new Intent(Intent.ACTION_EDIT);
intent.setType("vnd.android.cursor.item/event");
   intent.putExtra("beginTime", cal.getTimeInMillis());
   intent.putExtra("allDay", true);
   intent.putExtra("rrule", "FREQ=YEARLY");
   intent.putExtra("endTime", cal.getTimeInMillis()+60*60*1000);
   intent.putExtra("title", “Test Event");

   startActivity(intent);
}
```

iOS: EventKit + UI

- Event Kit
  - Objective-C Framework
    - Programmatic access to the calendar
- Event Kit UI
  - Objective-C Framework
  - View controllers for accessing the calendar

Programmatic Access

- Create a single (shared) event store
  - E.g., as property in the AppDelegate
- Access the event database
  - Fetch events via NSPredicate

```
NSArray *events = [eventStore eventsMatchingPredicate:predicate];
```

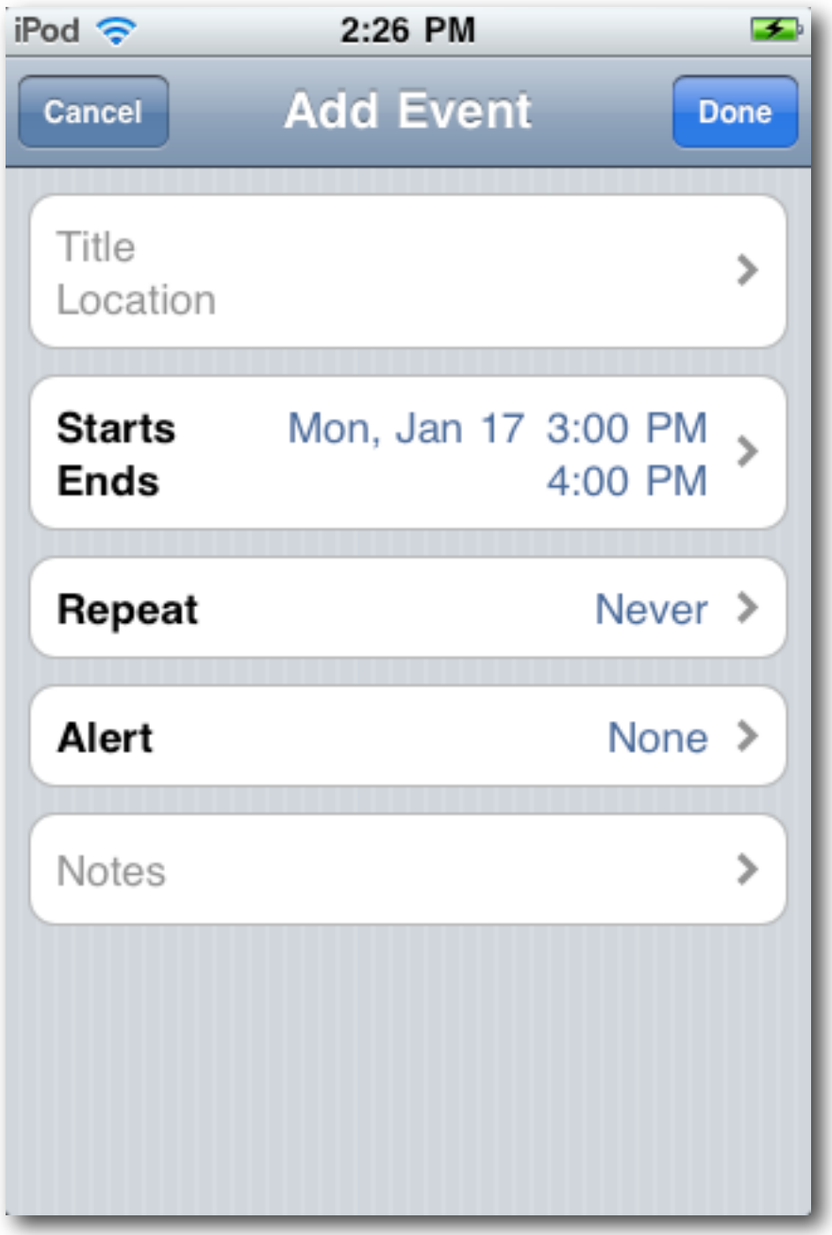
  - Create & configure events

```
EKEvent *event = [EKEvent eventWithEventStore:self.eventStore];
event.title = @"iPhone Course";
...
[self.eventStore saveEvent:event span:EKSpanThisEvent error:&error];
```

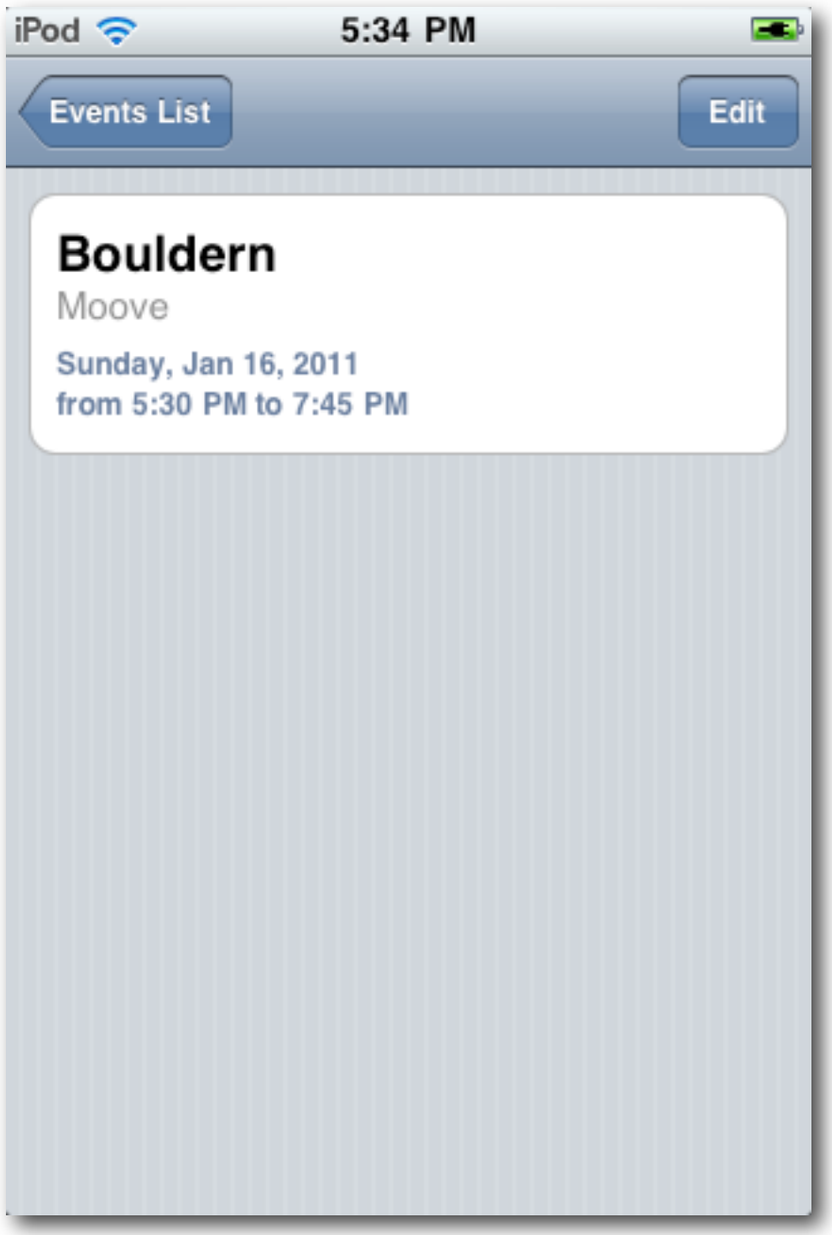
Fetching Events (Example)

```
- (NSArray *)fetchEventsForToday {

   NSDate *startDate = [NSDate date];

   // endDate is 1 day = 60*60*24 seconds = 86400 seconds from startDate
   NSDate *endDate = [NSDate dateWithTimeIntervalSinceNow:86400];

   // Create the predicate. Pass it the default calendar.
   NSArray *calendarArray = [NSArray arrayWithObject:defaultCalendar];
   NSPredicate *predicate = [self.eventStore
     predicateForEventsWithStartDate:startDate
                             endDate:endDate
                           calendars:calendarArray];

   // Fetch all events that match the predicate.
   NSArray *events = [self.eventStore eventsMatchingPredicate:predicate];

   return events;
}
```

Event Kit UI

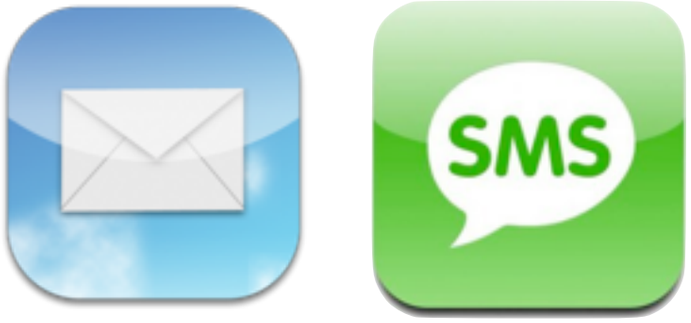


EKEventViewController



EKEventEditViewController

Email



iOS: Message UI

- Compose email / SMS within your application
  - Your application is not quit
- Composition via View Controllers
  - No programmatic email / sms delivery!
  - Email: MFMailComposeViewController
  - SMS: MFMessageComposeViewController

Compose an Email

```
// Verify that we can send mails
if (![MFMailComposeViewController canSendMail])
   return;

// Create a Mail Compose View Controller
MFMailComposeViewController *composeViewController =
   [[MFMailComposeViewController alloc] init];

// Set the delegate
composeViewController.mailComposeDelegate = self;

// Configure Email
[composeViewController setToRecipients:[NSArray
   arrayWithObject:toTextField.text]];
[composeViewController setSubject::@"Hello from Aachen!"];

// Present the Mail Compose VC
[self presentModalViewController:composeViewController
   animated:YES];
[composeViewController release];
```

Android: Email via Intent

```
Intent emailIntent = new Intent(
  android.content.Intent.ACTION_SEND);

emailIntent.setType("plain/text");

emailIntent.putExtra(android.content.Intent.EXTRA_EMAIL,
  new String[] { address.getText().toString() });

emailIntent.putExtra(android.content.Intent.EXTRA_SUBJECT,
  subject.getText());

emailIntent.putExtra(android.content.Intent.EXTRA_TEXT,
  emailtext.getText());

Email.this.startActivity(Intent.createChooser(emailIntent,
  "Send mail..."));
```

Camera

Android: Demo

iOS: Image Picker

- View controller
  - UIImagePickerController
- Source Types:
  - Camera (front or back)
  - Photo library
- Supports user or programmatic capture

Example

```
- (IBAction)takePicture {
   IImagePickerController *picker = [[UIImagePickerController alloc]
     init];
   picker.delegate = self;
   picker.sourceType = UIImagePickerControllerSourceTypeCamera;
   [self presentModalViewController:picker animated:YES];
   [picker release];
}

// Delegate Method: image picker is done
- (void)imagePickerController:(UIImagePickerController *)picker
didFinishPickingMediaWithInfo:(NSDictionary *)info {

   UIImage *image = [info
     valueForKey:UIImagePickerControllerOriginalImage];

   // do something...

   [self dismissModalViewControllerAnimated:YES];
}
```

User Preferences & Settings

Preferences and Settings

- NSUserDefaults
  - Automatically stores settings for the user
  - Write:

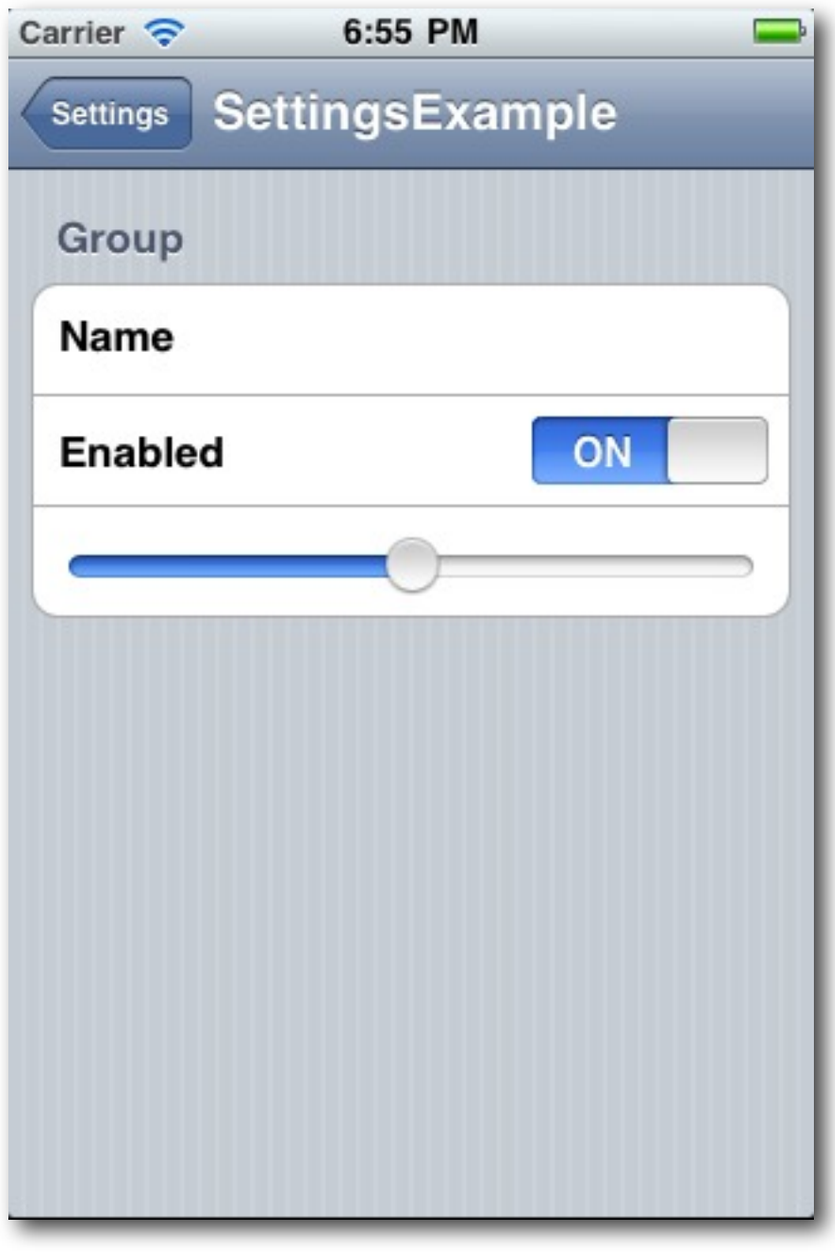
```
[[NSUserDefaults standardUserDefaults] setValue:name
  forKey:@"name"];
```

  - Read:

```
NSString *name = [[NSUserDefaults standardUserDefaults]
  valueForKey:@"name"];
```

Settings Bundle

- Define the settings UI of your application
  - Add Resource `Settings Bundle`
    - Configure `Root.plist` inside the bundle
      - Available Types: Group, Text Field, Toggle Switch, Slider
- Settings are stored in User Defaults



Background Execution & Notifications

iOS: Background Task Types

- By default applications are suspended when the user presses the home button
- Background tasks are declared in Info.plist:
  - `audio`: application keeps playing audio
  - `location`: application keeps receiving location triggers
  - `voip`: application provides voice-over-ip services
- Tasks can also request extra time to complete a complex task or schedule notifications

Complete a Complex Task

```
// Application Delegate Method
- (void)applicationDidEnterBackground:(UIApplication *)application
{
   // request background task
   UIBackgroundTaskIdentifier taskId = [application
     beginBackgroundTaskWithExpirationHandler:^(void)
   {
      NSLog(@"We expired!");
   }];

   // perform your complex task...

   // we are done
   [application endBackgroundTask:taskId];
}
```

Schedule a Notification

```
- (void)scheduleNotification
{
   UIApplication *application = [UIApplication sharedApplication];

   // Clear out any old notifications before scheduling a new one.
   if([[application scheduledLocalNotifications] count] > 0) {
      [application cancelAllLocalNotifications];
   }

   // Create a new notification to be fired in 5 seconds
   UILocalNotification* alarm = [[UILocalNotification alloc] init];
   alarm.fireDate = [NSDate dateWithTimeIntervalSinceNow:5.0];
   alarm.repeatInterval = 0;
   alarm.soundName = @"alarmsound.caf";
   alarm.alertBody = @"Time to wake up!";
   alarm.applicationIconBadgeNumber = 1;

   [application scheduleLocalNotification:alarm];
   [alarm release];
}
```

Track the User's Location

- Track Significant Location Updates
  - only major updates activate the application
  - saves battery

```
[manager startMonitoringSignificantLocationChanges];
```

- Track All Location Updates
  - Add `location` to `Required background modes` in Info.plist
  - Location updates from CLLocationManager are executed in the background

Android: Background Service

- Longer-running operation
- (Nearly) not interacting with the user
- Supplying functionality to other applications
- To be declared in the manifest

Handling Services

- Starting a service:

```
startService(new Intent(this, MyService.class));
```

- Stopping a service:

```
stopService(new Intent(this, MyService.class));
```

Service Class

```
public class MyService extends Service {
    MediaPlayer player;
    …

    @Override
    public void onCreate() {
        player = MediaPlayer.create(this, R.raw.mechanism);
        player.setLooping(false);
    }

    @Override
    public void onDestroy() {
        player.stop();
    }

    @Override
    public void onStart(Intent intent, int startid) {
        player.start();
    }
}
```

Notifications

```
NotificationManager nManager = (NotificationManager) this
    .getSystemService(Context.NOTIFICATION_SERVICE);

Intent i = new Intent(this, de.test.service.main.class);

Notification notification = new Notification(R.drawable.icon, this
    .getString(R.string.app_name), System.currentTimeMillis());

notification.setLatestEventInfo(this, this
    .getString(R.string.app_name), this
    .getString(R.string.app_name), PendingIntent.getActivity(
        this, 0, i, 0));

notification.flags |= Notification.FLAG_AUTO_CANCEL;

nManager.notify(0, notification);
```

Notifications