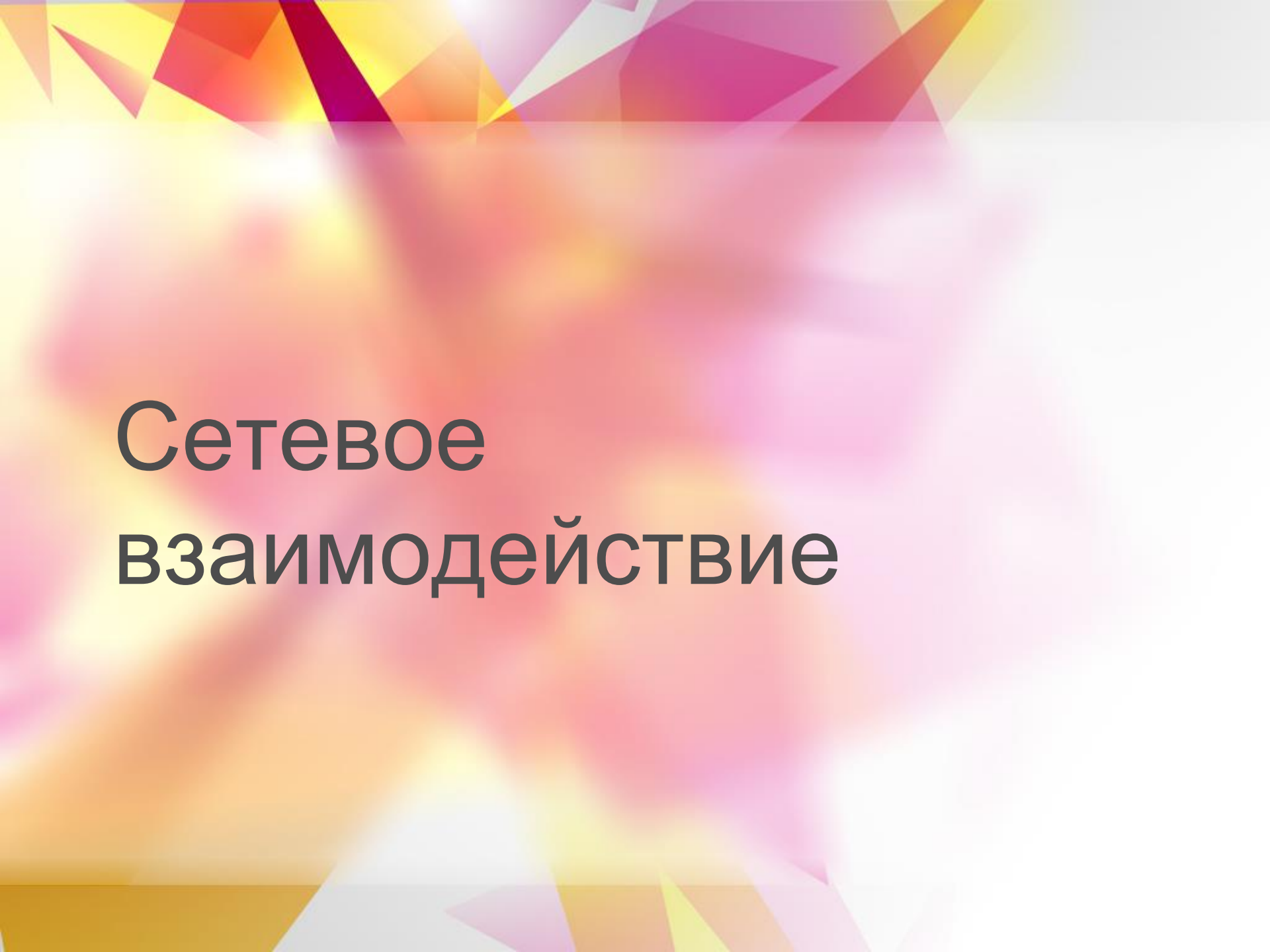


Мобильная разработка

Пирская Любовь Владимировна,
к.т.н., доцент кафедры МОП ЭВМ
lpirskaya@sfedu.ru

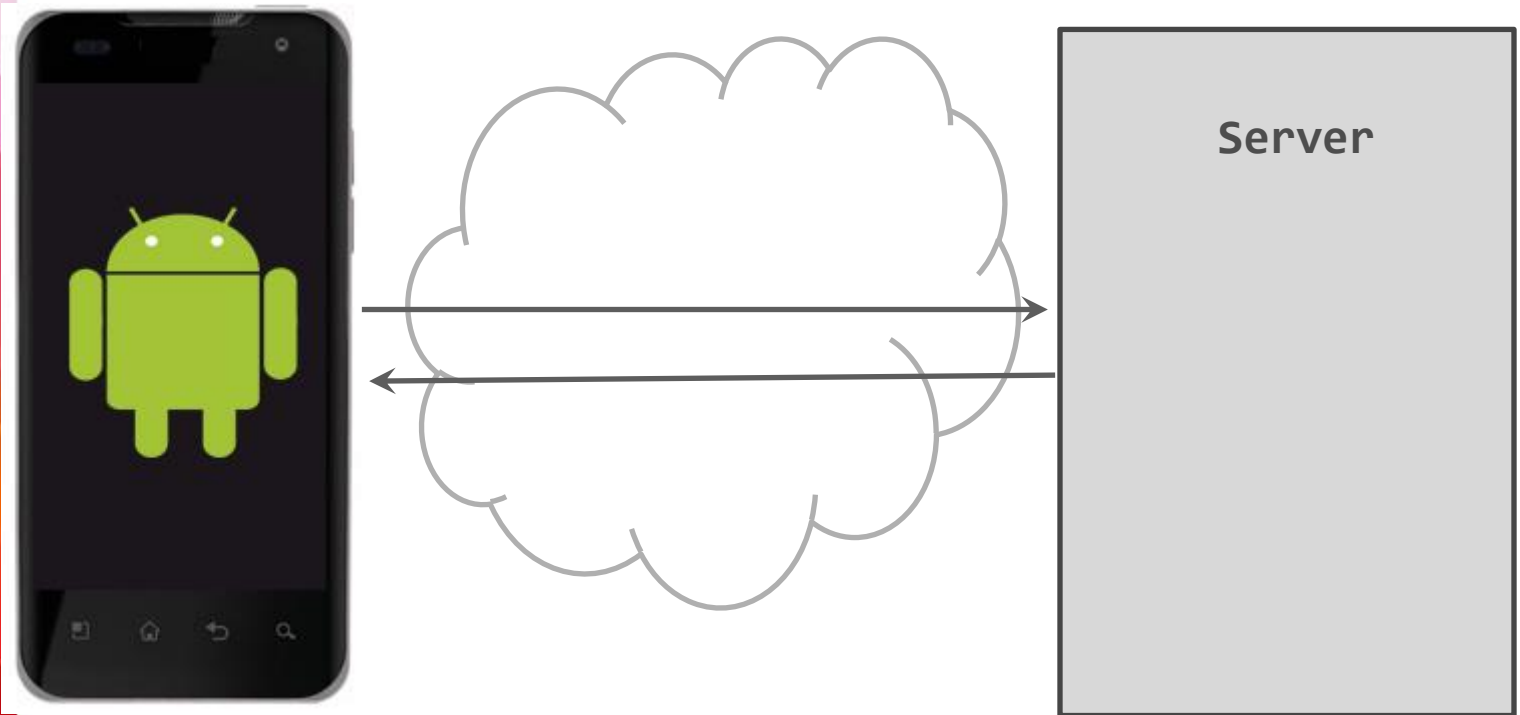


Сетевое взаимодействие

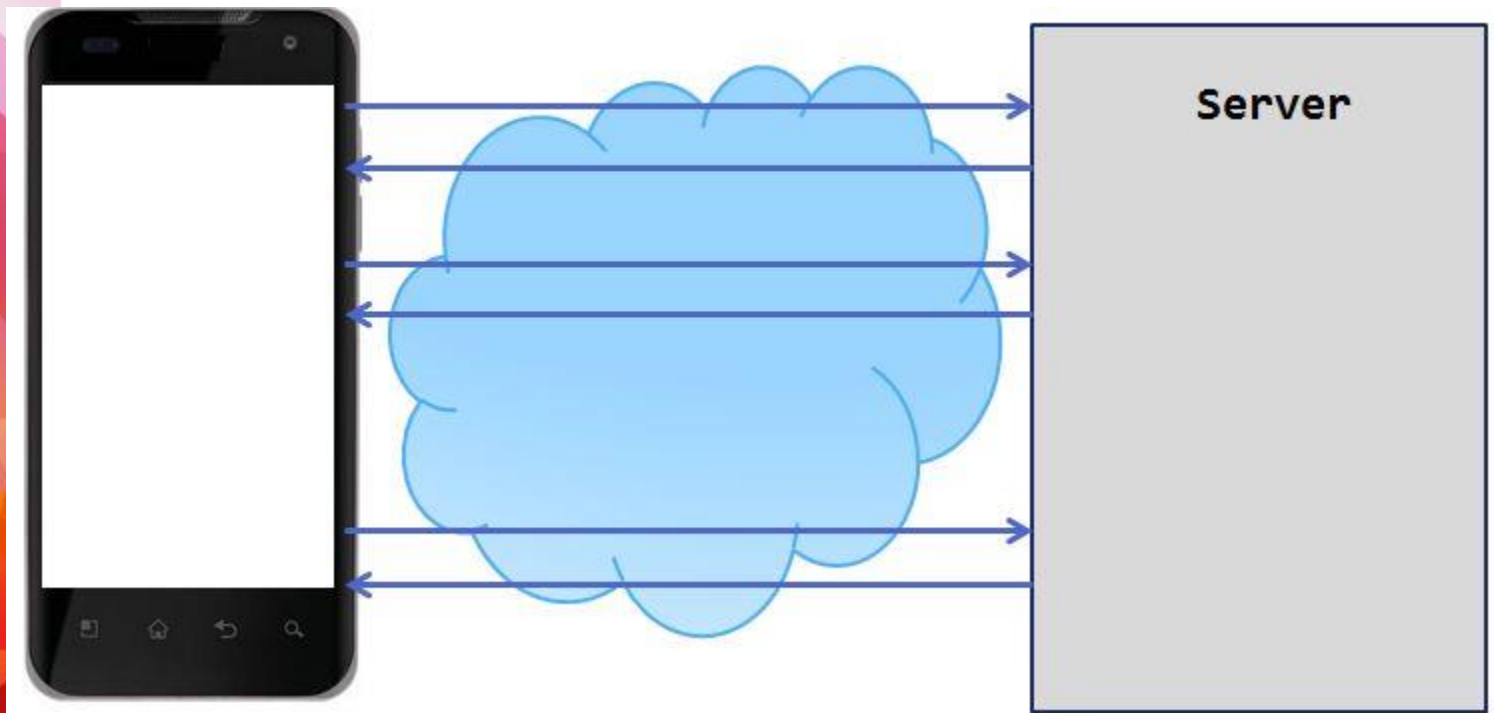
О чем нужно помнить

- **Трафик**
- **Батарейка**
- **Безопасность**

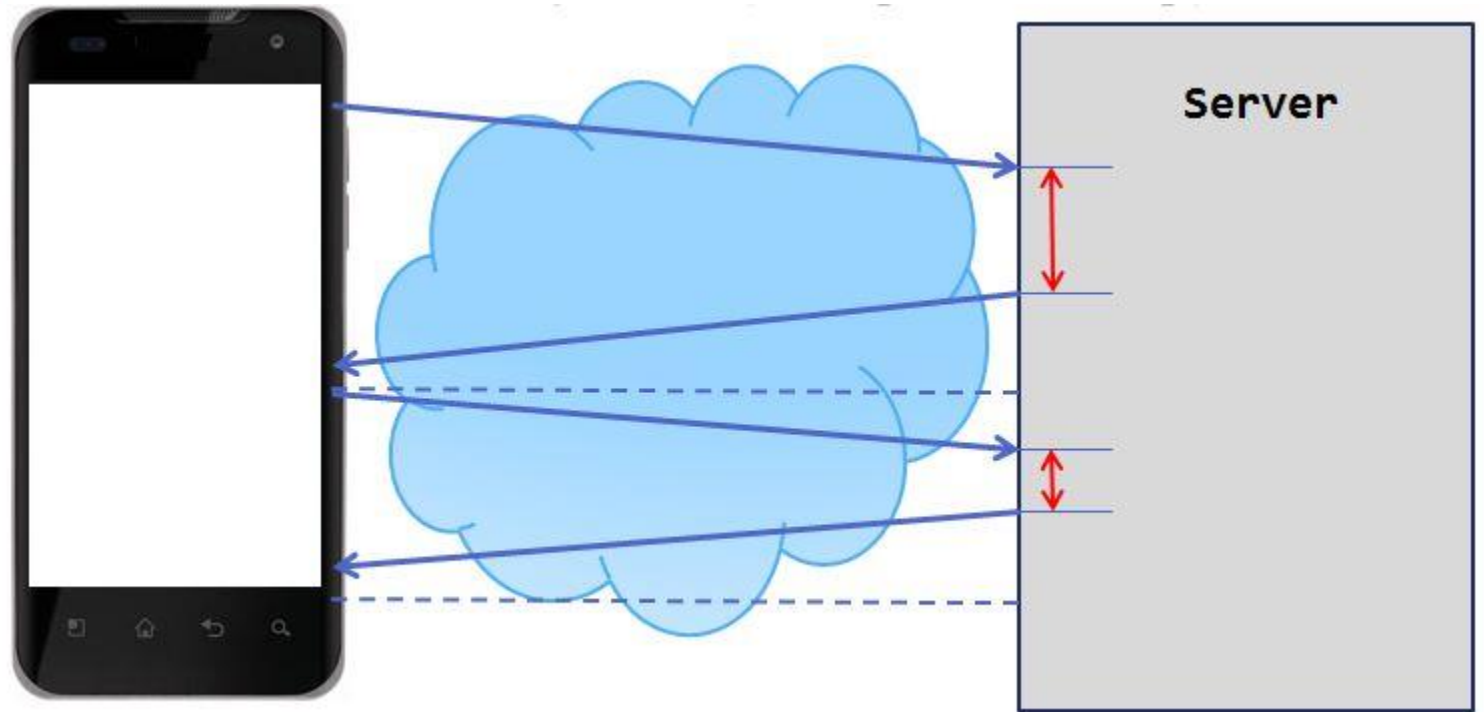
Серверное взаимодействие



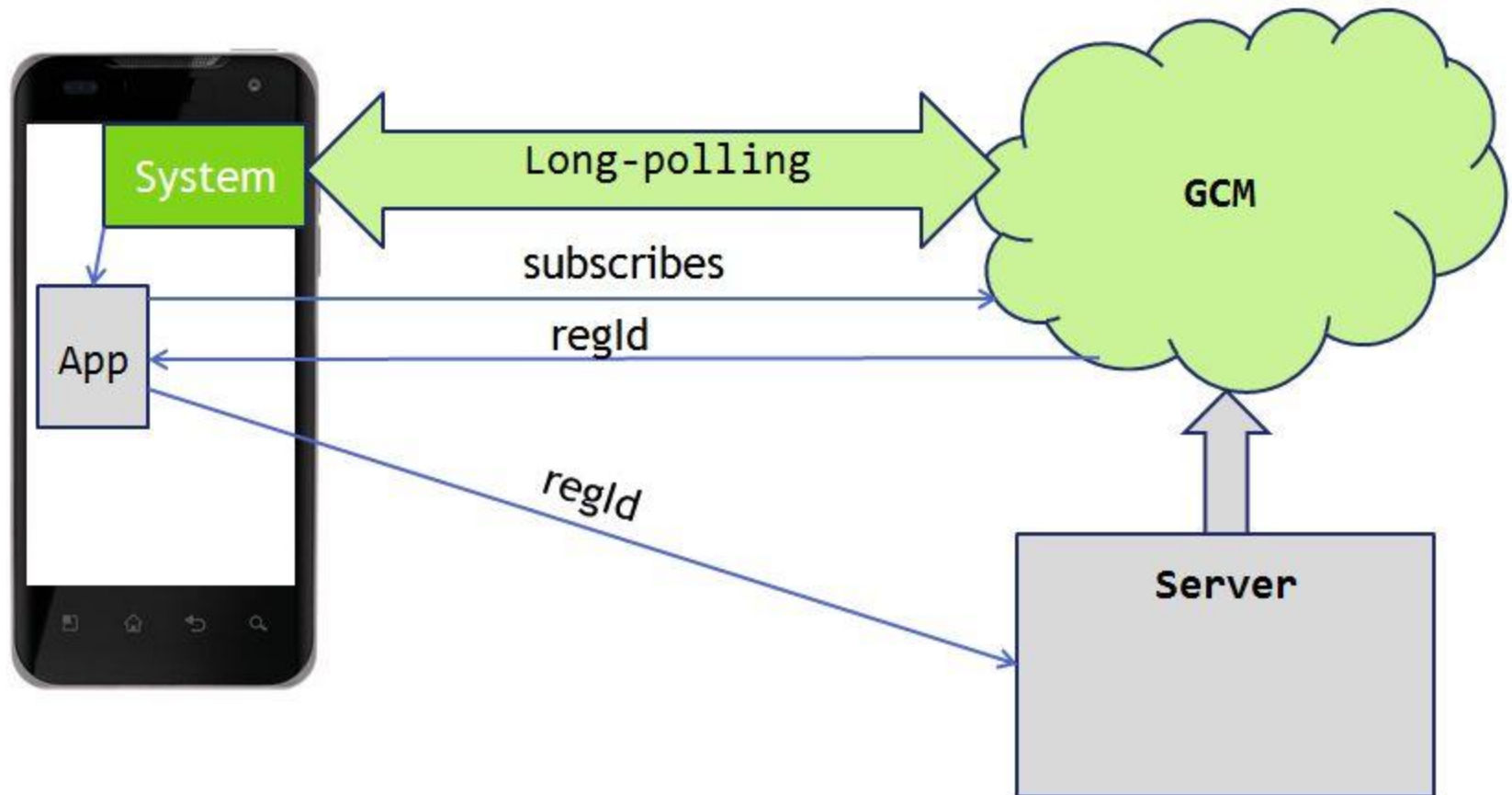
Polling



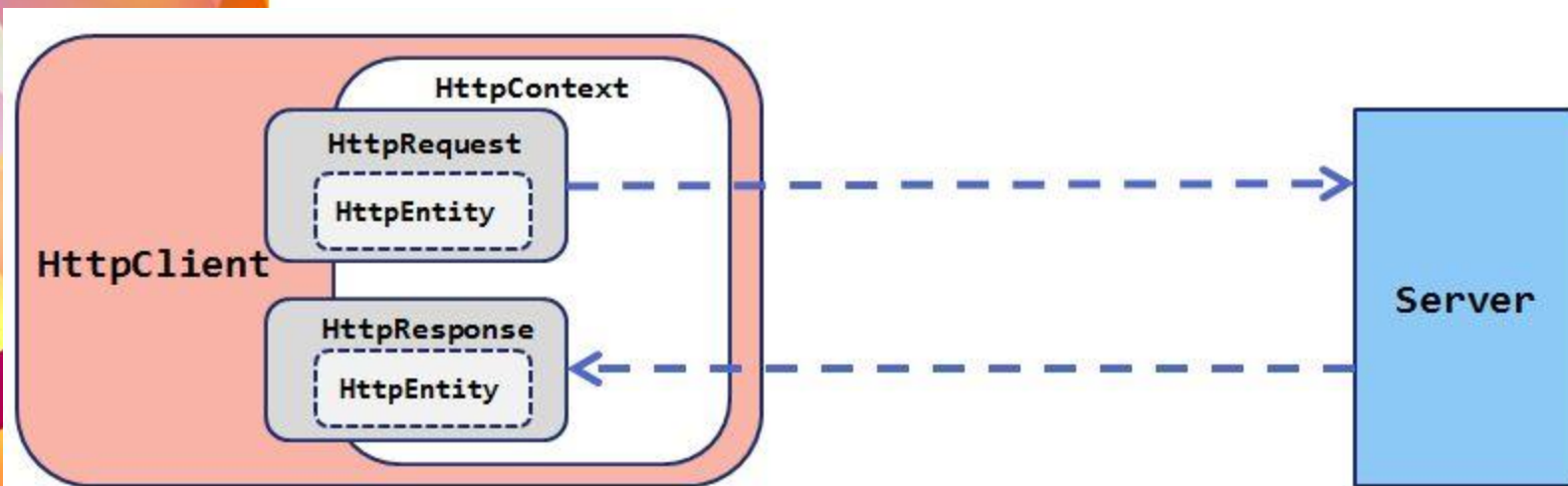
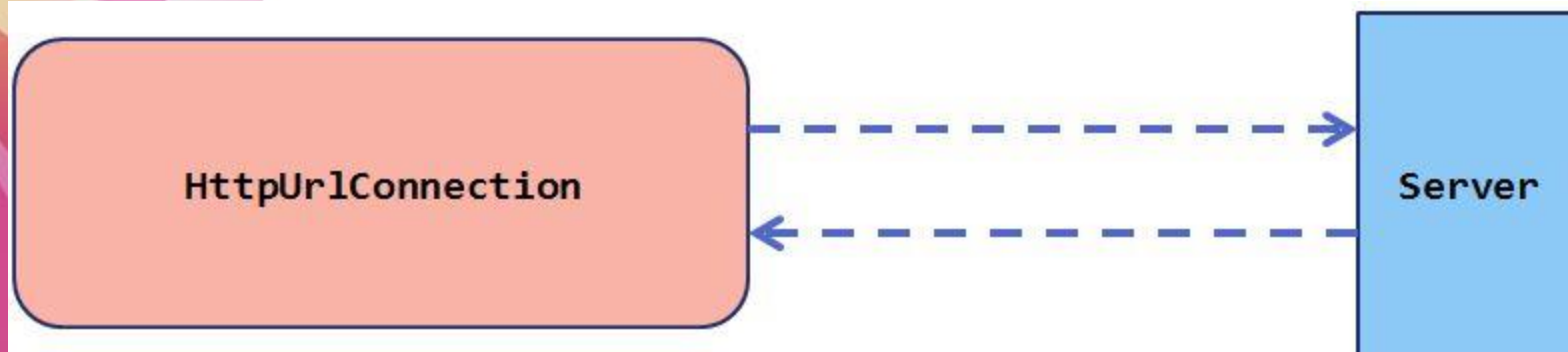
Long-Polling



Push notifications



Варианты взаимодействия



URLConnection

- **Lightweight**
- **Один класс**
- **Прост в использовании**
- **Поддержка всего, что нужно**
- **В поздних версиях gзір из коробки, кеширование, авторизация, IPv6 и проч.**
- **Рекомендован Google**
- **Баги**

URLConnection

```
URLConnection connection = null;  
try {  
    URL url = new  
URL("http://example.com");  
    connection = (URLConnection)  
url.openConnection();
```

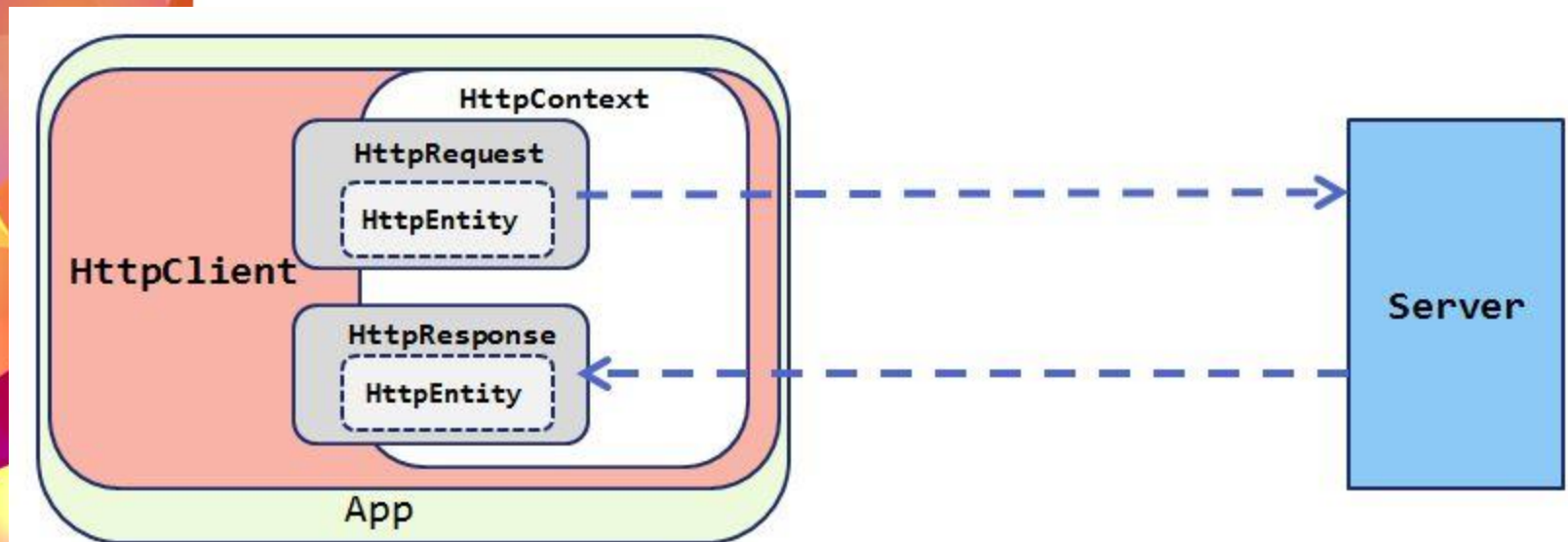
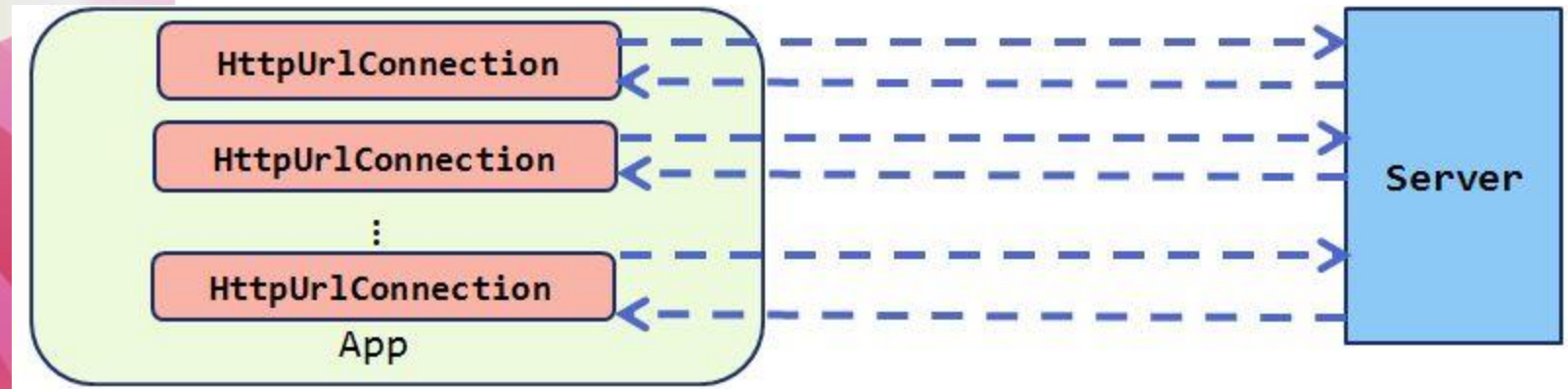
URLConnection

```
URLConnection connection = null;  
try {  
    URL url = new  
URL("http://example.com");  
    connection = (URLConnection)  
url.openConnection();  
    connection.setRequestMethod("GET");  
  
connection.setRequestProperty("Accept"  
, "text/plain");
```

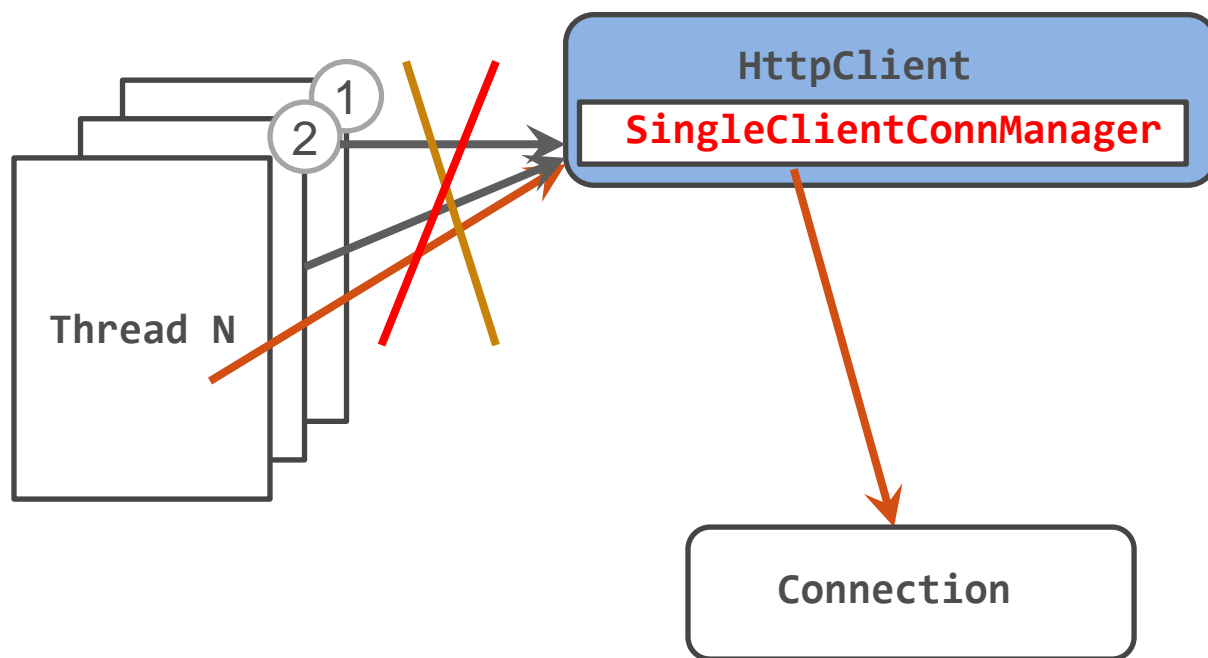
URLConnection

```
URLConnection connection = null;  
try {  
    URL url = new  
URL("http://example.com");  
    connection = (URLConnection)  
url.openConnection();  
    connection.setRequestMethod("GET");  
  
connection.setRequestProperty("Accept"  
, "text/plain");  
    connection.connect();
```

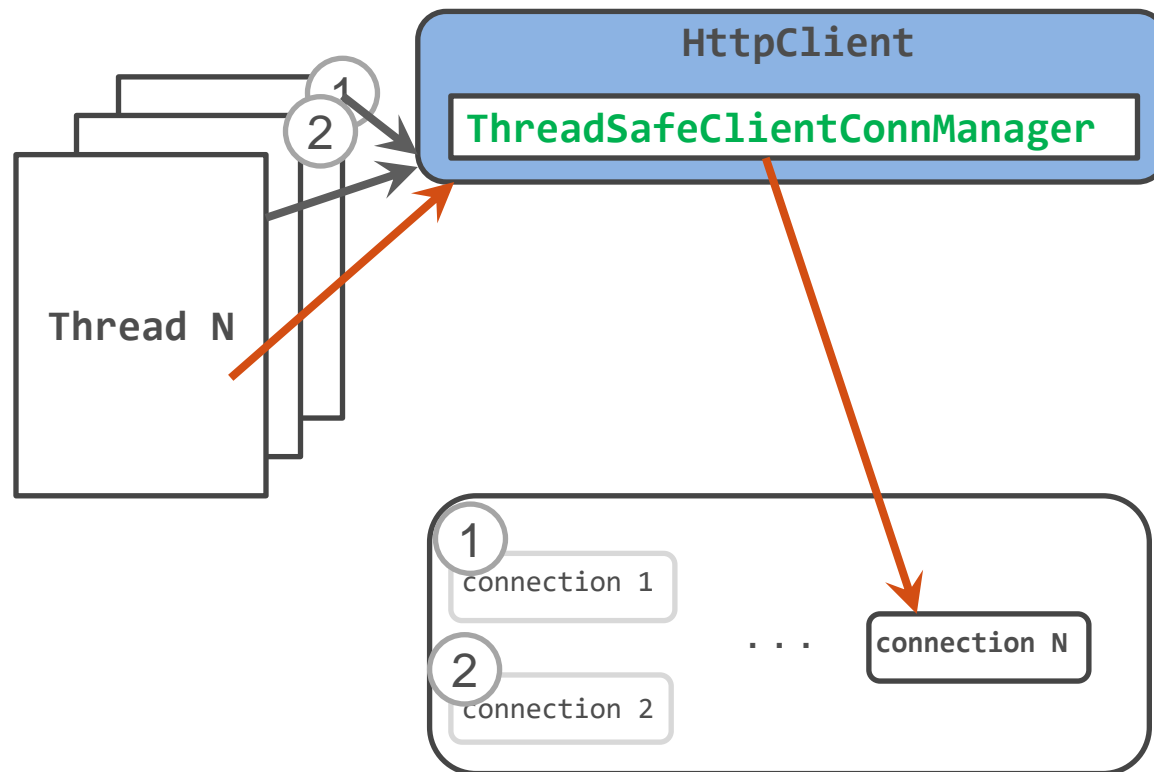
Разница



Потокобезопасность



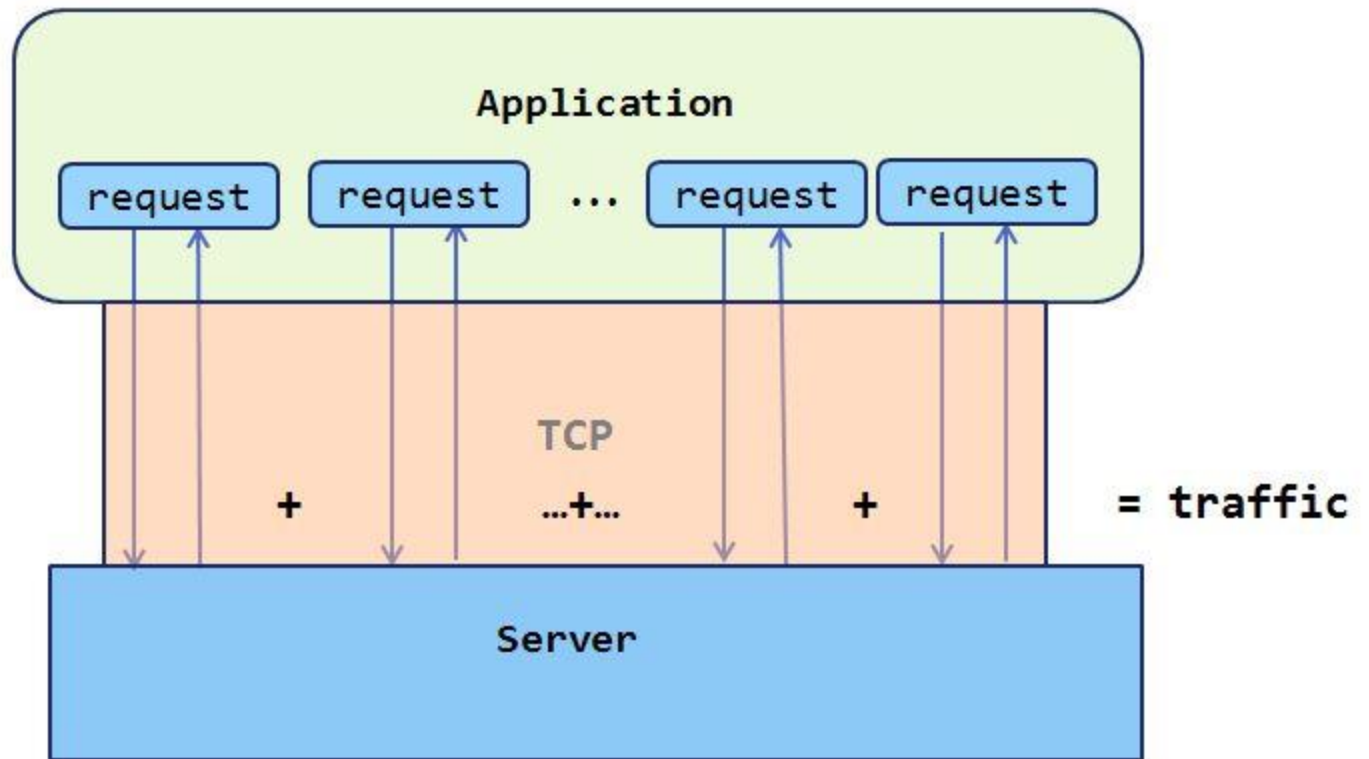
Потокобезопасность



Потокобезопасность

```
static {
    ...
    HttpParams params = new BasicHttpParams();
    ConnManagerParams.setMaxTotalConnections(params,
10);
    ConnManagerParams.setMaxConnectionsPerRoute(params,
        new ConnPerRoute() {
            @Override
            public int getMaxForRoute(HttpRoute route) {
                return 5;
            }
        });
    ThreadSafeClientConnManager cm =
        new ThreadSafeClientConnManager(params,
schemeRegistry);
    httpClient = new DefaultHttpClient(cm, params);
}
```

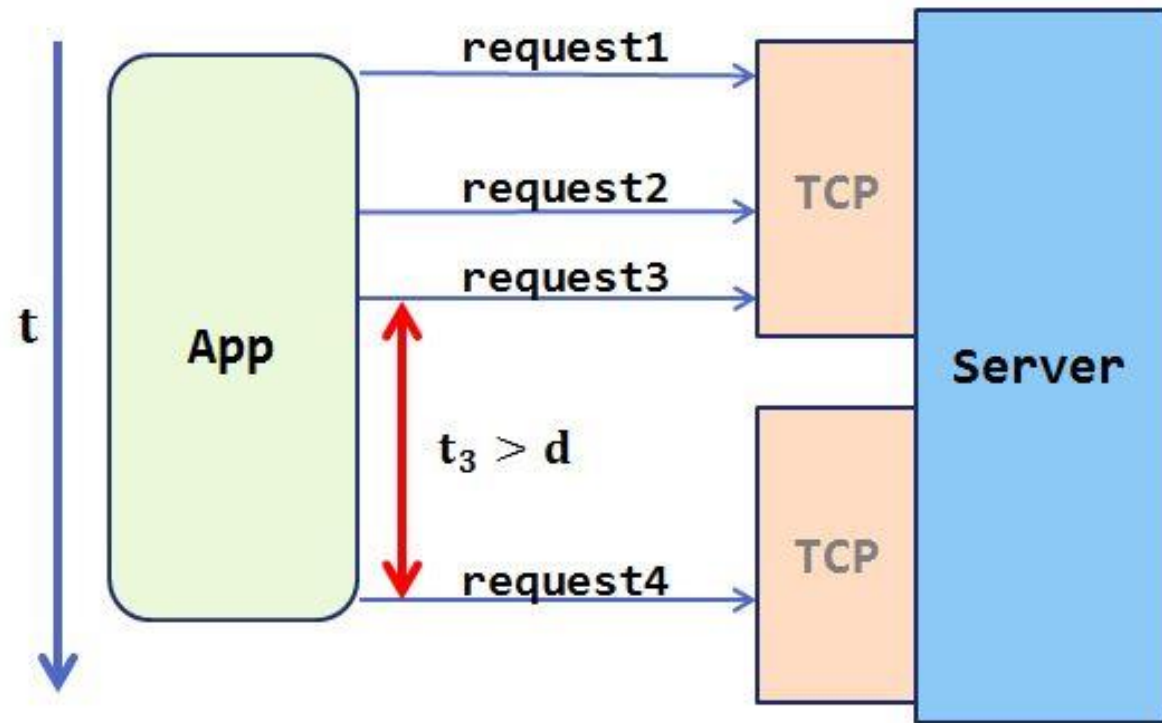
Keep-alive



Keep-alive

Номер запроса	Время(ms) KeepAlive = false	Время(ms) KeepAlive = true
1	2098	2023
2	2157	1604
3	2037	1698
4	2096	1774
5	1944	1173
6	2055	1573
7	1865	1683
8	2119	1670
9	1986	1666
10	1965	1541
	≈2032,2	≈1700,5

Keep-alive duration



t – time

d – keep alive duration

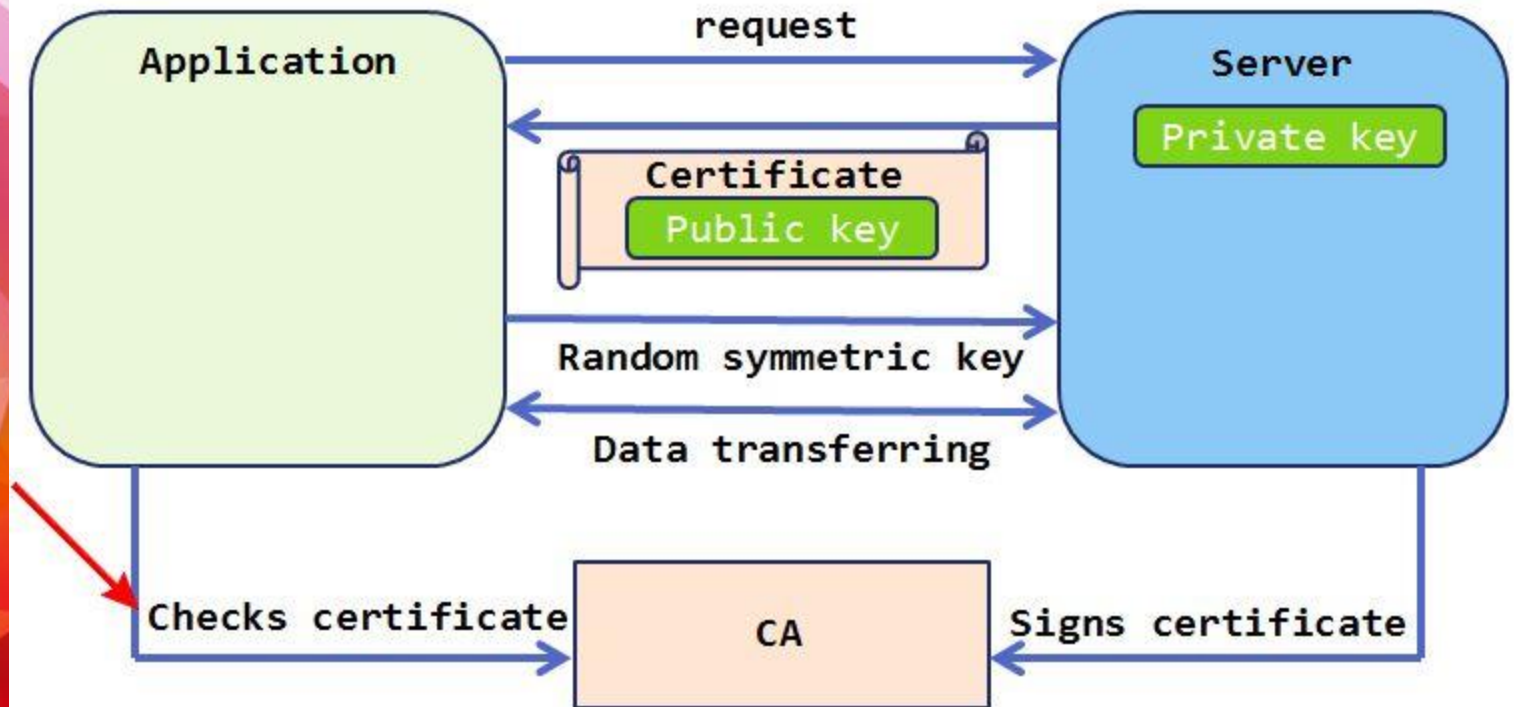
Keep-alive prior Froyo

```
if (Build.VERSION.SDK_INT <=
    Build.VERSION_CODES.FROYO){
    System.setProperty("http.keepAlive",
        "false");
}
```


GZip

```
...
InputStream is = response.getEntity().getContent();
Header contentEncoding =
    response.getFirstHeader("Content-Encoding");
if (contentEncoding != null
    &&
    contentEncoding.getValue().equalsIgnoreCase("gzip"))
{
    is = new GZIPInputStream(is);
}
...
...
InputStream is = connection.getInputStream();
String contentEncoding =
    connection.getContentEncoding();
if ("gzip".equalsIgnoreCase(contentEncoding)) {
    is = new
    GZIPInputStream(connection.getInputStream());
}
...
```

Https



Https

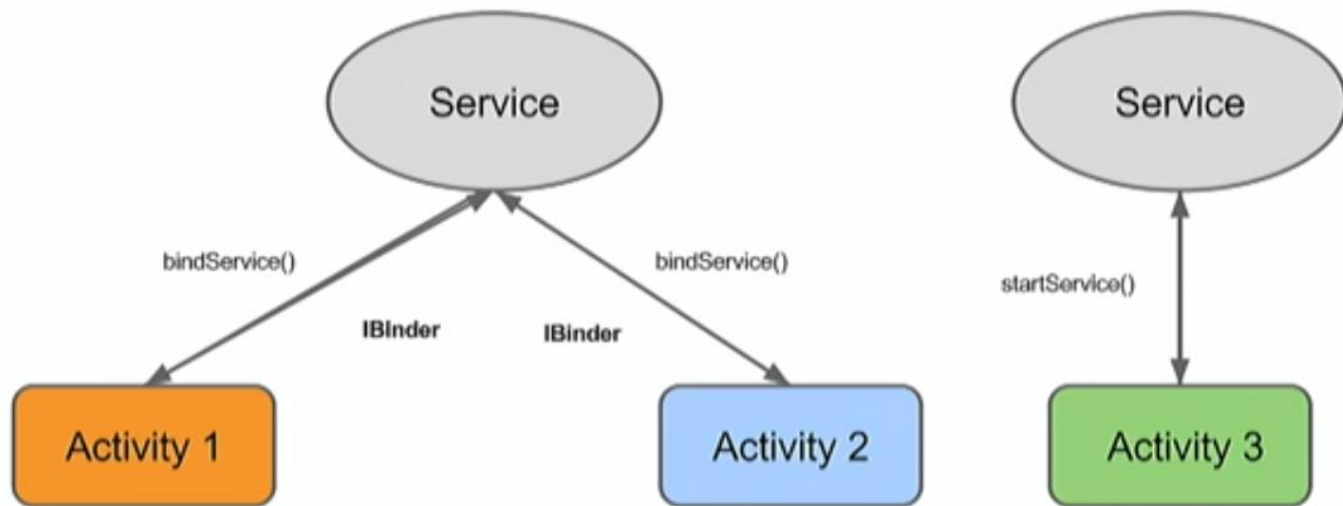
- KeyChain API на платформах ≥ 4.0
- < 4.0 – создавать локальное хранилище ключей
- Доверять всем сертификатам



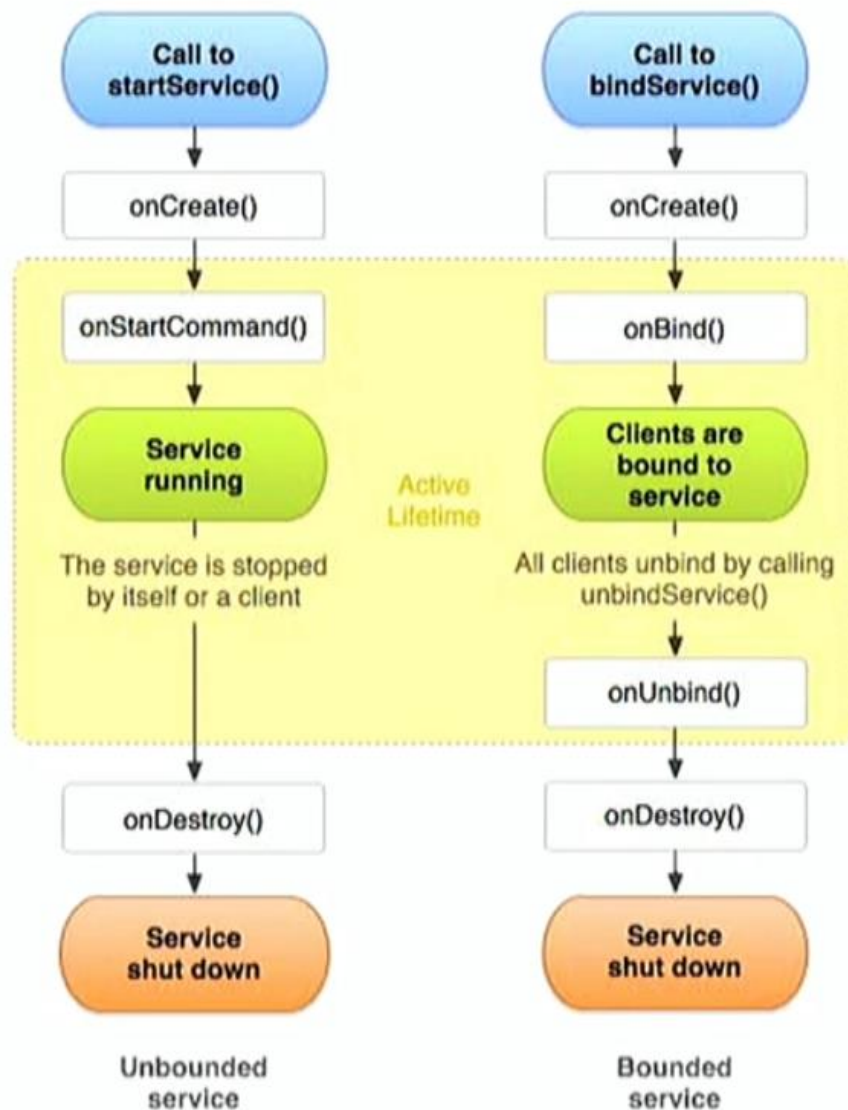
Сервисы

Сервисы

- `Context.startService();`
- `Context.bindService().`



Жизненный цикл сервисов



Создание сервиса. Пример

```
public class PlayService extends Service {

    private MediaPlayer mPlayer;

    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override
    public void onCreate() {
        super.onCreate();
        Toast.makeText(this, "Служба создана",
            Toast.LENGTH_SHORT).show();
        mPlayer = MediaPlayer.create(this, R.raw.flower_romashka);
        mPlayer.setLooping(false);
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        Toast.makeText(this, "Служба запущена",
            Toast.LENGTH_SHORT).show();
        mPlayer.start();
        return super.onStartCommand(intent, flags, startId);
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        Toast.makeText(this, "Служба остановлена",
            Toast.LENGTH_SHORT).show();
        mPlayer.stop();
    }
}
```

Создание сервиса. Пример

- Зарегистрируем службу в манифесте

```
<service
    android:enabled="true"
    android:name=".PlayService">
</service>
```

- Запуск и остановка сервиса

```
// запуск службы
btnStart.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        // используем явный вызов службы
        startService(
            new Intent(MainActivity.this, PlayService.class));
    }
});

// остановка службы
btnStop.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        stopService(
            new Intent(MainActivity.this, PlayService.class));
    }
});
```

Создание сервиса. Пример

- Если необходимо, чтобы сервис запускался и после перезагрузки устройства, то следует создать приёмник широковещательных сообщений и запустить в нём сервис

```
public class BootBroadcast extends BroadcastReceiver {  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        context.startService(new Intent(context, TestService.class));  
    }  
}
```

- Приёмник регистрируется в манифесте с именем действия **BOOT_COMPLETED**

```
<receiver android:name=".BootBroadcast">  
    <intent-filter >  
        <action android:name="android.intent.action.BOOT_COMPLETED"/>  
    </intent-filter>  
</receiver>
```