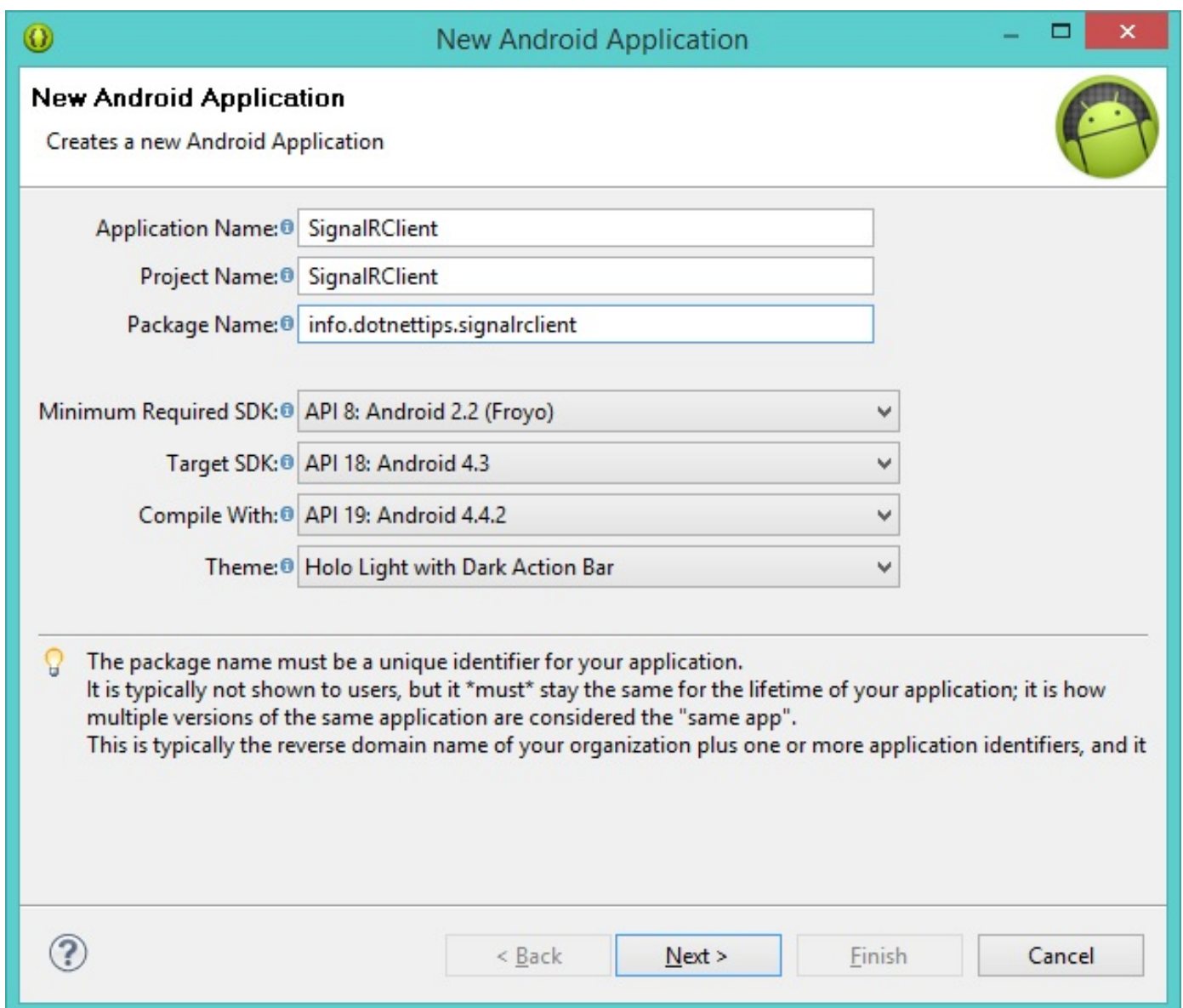


همانطور که مطلع هستید، بخش سورس باز مایکروسافت برای برنامه‌نویس‌های جاوا نیز [SDK](#) ی جهت استفاده از SignalR ارائه کرده است. در [اینجا](#) می‌توانید مخزن کد آن را در گیت‌هاب مشاهده کنید. هنوز مستنداتی برای این SDK به صورت قدم به قدم ارائه نشده است. لازم به ذکر است که مراجعه به قسمت‌های نوشته شده در [اینجا](#) نیز می‌تواند منبع خوبی برای شروع باشد. در ادامه نحوه استفاده از این SDK را با هم بررسی خواهیم کرد. ابتدا در سمت سرور یک Hub ساده را به صورت زیر تعریف می‌کنیم:

```
public class ChatHub : Hub
{
    public void Send(string name, string message)
    {
        Clients.All.messageReceived(name, message);
    }
}
```

برای سمت کلاینت نیز یک پروژه Android Application داخل Eclipse به صورت زیر ایجاد می‌کنیم:



New Android Application
Creates a new Android Application

Application Name:

Project Name:


Package Name:


Minimum Required SDK:

Target SDK:

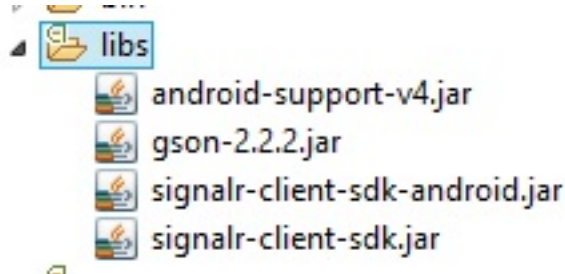
Compile With:

Theme:

 The package name must be a unique identifier for your application. It is typically not shown to users, but it *must* stay the same for the lifetime of your application; it is how multiple versions of the same application are considered the "same app". This is typically the reverse domain name of your organization plus one or more application identifiers, and it



خوب، برای استفاده از SignalR در پروژه‌ی ایجاد شده باید کتابخانه‌های زیر را به درون پوشه libs اضافه کنیم، همچنین باید ارجاعی به کتابخانه [Gson](#) نیز داشته باشیم.



قدم بعدی افزودن کدهای سمت کلاینت برای SignalR می‌باشد. دقت داشته باشید که کدهایی که در ادامه مشاهده خواهید کرد دقیقاً مطابق دستورالعمل‌هایی است که [قبلاً](#) مشاهده کرده‌اید. برای اینکار داخل کلاس MainActivity.java کدهای زیر را اضافه کنید:

```
Platform.loadPlatformComponent( new AndroidPlatformComponent() );
HubConnection connection = new HubConnection(DEFAULT_SERVER_URL);
HubProxy hub = connection.createHubProxy("ChatHub");
connection.error(new ErrorCallback() {

    @Override
    public void onError(final Throwable error) {
        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_LONG).show();
            }
        });
    }
});
hub.subscribe(new Object() {
    @SuppressWarnings("unused")
    public void messageReceived(final String name, final String message) {

        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), name + ": " + message,
                Toast.LENGTH_LONG).show();
            }
        });
    }
});
connection.start()
.done(new Action<Void>() {

    @Override
    public void run(Void obj) throws Exception {
        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), "Done Connecting!", Toast.LENGTH_LONG).show();
            }
        });
    }
});
connection.received(new MessageReceivedHandler() {
    @Override
    public void onMessageReceived(final JsonElement json) {
        runOnUiThread(new Runnable() {
            public void run() {
                JsonObject jsonObject = json.getAsJsonObject();
                JsonArray jsonArray = jsonObject.getAsJsonArray("A");
                Toast.makeText(getApplicationContext(), jsonArray.get(0).getString() + ": " +
                jsonArray.get(1).getString(), Toast.LENGTH_LONG).show();
            }
        });
    }
});
```

```
    }
});
```

همانطور که مشاهده می‌کنید توسط قطعه کد زیر SKD مربوطه در نسخه‌های قدیمی اندروید نیز بدون مشکل کار خواهد کرد:

```
Platform.loadPlatformComponent( new AndroidPlatformComponent() );
```

در ادامه توسط متد createHubProxy ارجاعی به هابی که در سمت سرور ایجاد کردیم، داده‌ایم:

```
HubProxy hub = connection.createHubProxy("ChatHub");
```

در ادامه نیز توسط یک روال رویدادگردان وضعیت اتصال را چک کرده‌ایم. یعنی در زمان بروز خطا در نحوه ارتباط یک پیام بر روی صفحه نمایش داده می‌شود:

```
connection.error(new ErrorCallback() {
    @Override
    public void onError(final Throwable error) {
        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_LONG).show();
            }
        });
    }
});
```

در ادامه نیز توسط کد زیر متد پویایی که در سمت سرور ایجاد کرده بودیم را جهت برقراری ارتباط با سرور اضافه کرده‌ایم:

```
hub.subscribe(new Object() {
    @SuppressWarnings("unused")
    public void messageReceived(final String name, final String message) {
        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), name + ": " + message,
                    Toast.LENGTH_LONG).show();
            }
        });
    }
});
```

برای برقراری ارتباط نیز کدهای زیر را اضافه کرده‌ایم. یعنی به محض اینکه با موفقیت اتصال با سرور برقرار شد پیامی بر روی صفحه نمایش ظاهر می‌شود:

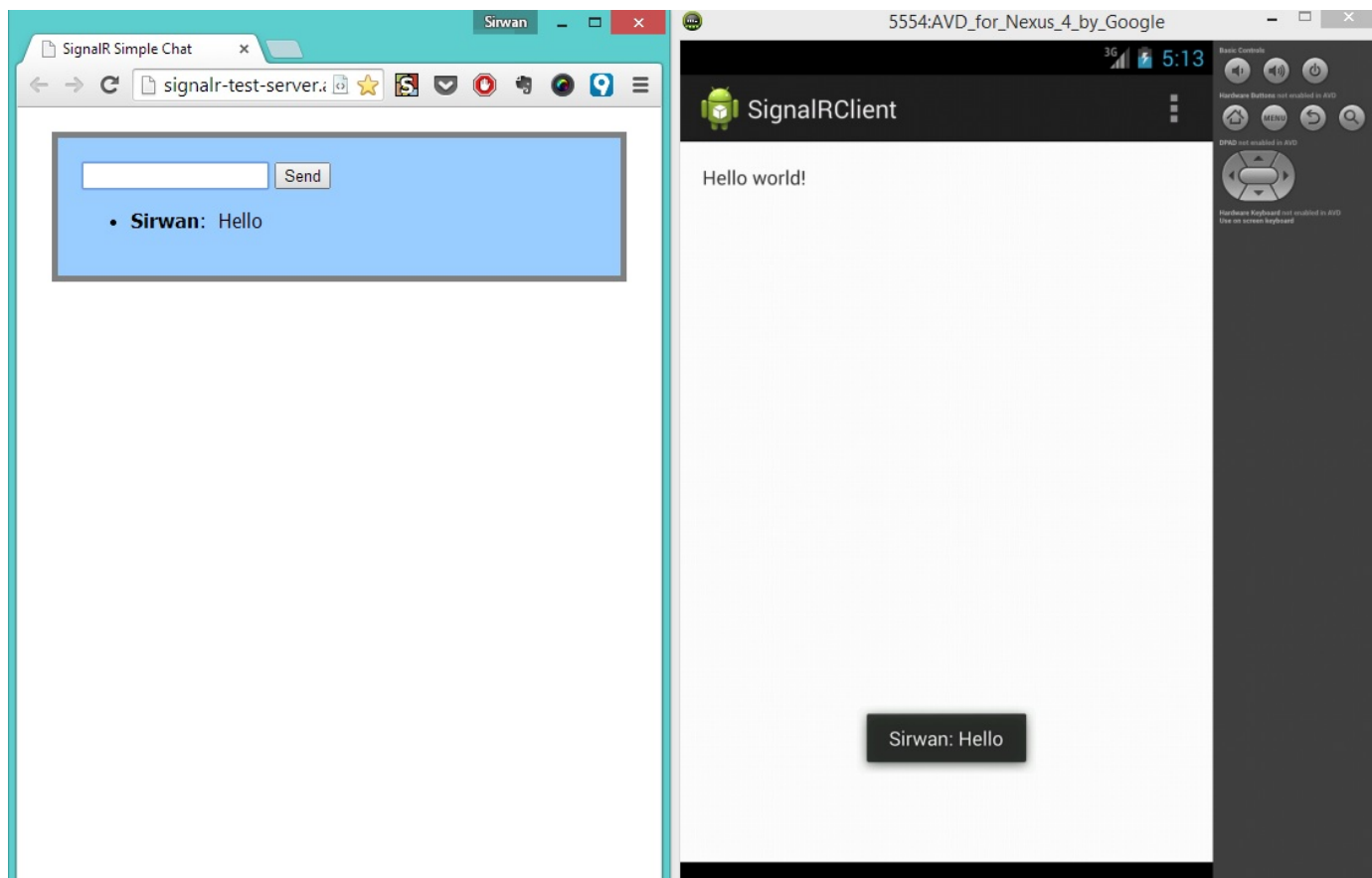
```
connection.start()
    .done(new Action<Void>() {
        @Override
        public void run(Void obj) throws Exception {
            runOnUiThread(new Runnable() {
                public void run() {
                    Toast.makeText(getApplicationContext(), "Done Connecting!", Toast.LENGTH_LONG).show();
                }
            });
        }
    });
```

در نهایت نیز برای نمایش اطلاعات دریافت شده کد زیر را نوشته‌ایم:

```
connection.receive(new MessageReceivedHandler() {
    @Override
    public void onMessageReceived(final JsonElement json) {
```

```
runOnUiThread(new Runnable() {
    public void run() {
        JSONObject jsonObject = json.getAsJsonObject();
        JSONArray jsonArray = jsonObject.getAsJSONArray("A");
        Toast.makeText(getApplicationContext(), jsonArray.get(0).getString() + ": " +
        jsonArray.get(1).getString(), Toast.LENGTH_LONG).show();
    }
});
});
```

همانطور که عنوان شد کدهای فوق دقیقاً براساس قواعد و دستورالعمل استفاده از SignalR در سمت کلاینت می‌باشد.



نظرات خوانندگان

نویسنده: رشیدیان
تاریخ: ۱۵:۴۶ ۱۳۹۳/۰۷/۲۱

ممنون - بسیار عالی
یک سؤال: آیا از این طریق میشه به همون قابلیت‌های Push Notification در GCM دست یافت؟
و اینکه چقدر این روش قابل اتکا هست؟

نویسنده: سیروان عقیفی
تاریخ: ۱۶:۵۱ ۱۳۹۳/۰۷/۲۱

دقیقاً یکی از استفاده‌هایی که برای خودم داره بحث Push Notification و ارسال پیام به کاربران متصل هست.

نویسنده: علی ساری
تاریخ: ۹:۴۵ ۱۳۹۴/۰۴/۰۳

با تشکر از مطلب خوبتون
در طی این مدت بازخوردهای شما از سیگنال آر در پروژ‌هایی که ازش استفاده کردید چطور بوده؟
چند برابر بقیه سرویس‌ها مثل gcm یا parse بار روی سرور میاره؟
و اینکه مزیت‌های سیگنال آر در مقایسه با این 2 سرویس چیه (البته parse که میدونم رایگان نیست)
ممنون

نویسنده: سیروان عقیفی
تاریخ: ۱۲:۲۰ ۱۳۹۴/۰۴/۰۳

یکی از مزایای استفاده از SignalR جهت ارسال push notification سفارشی‌سازیه، یعنی شما با استفاده از قابلیت مثل [backplan](#) به راحتی می‌تونید message‌ها رو به سرورهای دیگه فوروارد کنید همچنین می‌تونید از یکسری [تکنیک](#) برای داشتن performance بهتر استفاده کنید به طور مثال کاهش سایز اشیاء سریالایز شده و ...
در مجموع انعطاف این روش خیلی بیشتره