Agenda

Module und Proxies

Schnittstellen JS <=> Java

Beispiel

Modules

```
var Sample = require(„ti.sample“);
var result = Sample.doSomething({
     city : „Berlin“,
     onload : onLoadFn
 });

Sample.addEventListener(„onload“,onloadFn);
```

Proxies

```
var Sample = require(„ti.sample“);

var A = Sample.createA();
var B = Sample.createB();

B.doSomethingWithA(A);
```

ViewProxies

```
var Sample = require(„ti.sample“);

var A = Sample.createA();
var win = Ti.UI.createWindow();

win.add(A);

A.method();
```

Use cases

Module only

Access to static native methods
(NDS-Browser, Badger, PrintManager, Tracking, RingtoneManager)

Proxy

Access to instances without views(WifiManager, Jsoup, WebScraper, WifiTransfer, IoT, NetCLients)

ViewProxy

Creating of own UIs
CompassView, SpectrumAnalyzer, FortuneWheel, Lottie

Module skeleton

```
@Kroll.module(name="Xxxx", id="xxxx.de")
public class XxxxModule extends KrollModule {

public XxxxModule() {
        super();
        // during require(„xxxx.de“)
     }

     @Kroll.onAppCreate
     public static void onAppCreate(TiApplication app) {
        // during start of app
        // maybe reading of tiapp.xml
     }

     // Methods
     @Kroll.method
     public String example() {
          return "hello world";
     }

     @Kroll.getProperty
     public String getExampleProp() {
          return "hello world";
     }


     @Kroll.setProperty
     public void setExampleProp(String value) {
          Log.d(LCAT, "set example property: " + value);
     }

}
```

Handling of arguments

Standard:

Key-Value-Store

JS: object
Java: KrollDict

Handling of arguments

```
public void importParams(@Kroll.argument(optional=true) Object obj) {
     String title;
    if (obj == null) {
       Log.d(LCAT,“importParams need a parameter“);
       return;
     }
     if (obj instance KrollDict) {
       KrollDict opts = (KrollDict) obj;
       if (opts.containsKeyAndNotNull(TiC.PROPERTY_TITLE)) {
         title = opts.getString(TiC.PROPERTY_TITLE);
       } else Log.w(LCAT,“title missing“);
     } else {
        Log.d(LCAT,“parameter of importParams must be an JS-object“);
       return;
     }

  }
```

Returns

```
KrollDict res = new KrollDict();
res.put(„title“,“My Title“);
res.put(„time“,(new Date).getTime());
res.put(„success“,Boolean.TRUE);
return res;
```

Callbacks & Events

```
private KrollFunction onLoadFn;

if (obj != null && obj instanceof KrollFunction) {
    onLoadFn = (KrollFunction) obj;
    } else Log.d(LCAT,“Need a callback to answer you question“)


// … later

KrollDict res = new KrollDict();
res.put(„success“,Boolean.TRUE);

if (onLoadFn != null) {
 onLoadFn.call(getKrollObject(),res);
}
if (hasListeners(„onload“))
 fireEvent(„onload“,res);
```

Lifecycles

```
public class SpectrumViewProxy extends TiViewProxy implements OnLifecycleEvent ;
```

```
@Override
public TiUIView createView(Activity activity) {
    ((TiBaseActivity) activity).addOnLifecycleEventListener(this);
    return new TiSpectrumView(this);
}
```

```
@Override
public void onPause(Activity activity) {
    handleStop(); // stop Animation
    super.onPause(activity);
}
```

Threading

```
@Kroll.method
public void stop() {
    if (TiApplication.isUIThread()) {
        handleStop();
    } else {
        TiMessenger.sendBlockingMainMessage(getMainHandler().obtainMessage(
                MSG_STOP));
    }
}
```

```
@Override
public boolean handleMessage(Message msg) {
    AsyncResult result = null;
    switch (msg.what) {
    case MSG_STOP: {
        result = (AsyncResult) msg.obj;
        handleStop();
        result.setResult(null);
        return true;
    }
    default: {
        return super.handleMessage(msg);
    }
    }
}
```

ViewProxy

Four stages:

SpectrumAnalyzerModule extends KrollModule
- Reference to require(„modulename“),
- holds constants,
- making static stuff

SpectrumViewProxy extends ViewProxy
- container for TiUIView
- handles lifecycle
- handles thread communication

TiSpectrumView extends TiUIView
- handles TiUI compatibility
- creates native view

SpectrumView extends View
- implements logic

ThirdParty - AAR

Problem:

- Modules supports only one res-Folder
- Thirdparty libraries uses own aars and aars from system.

Solution:

- Using a patch for module build and app build
- Two new folders in module: aars + aaa-jars
- module patch copies content of aars
- app patch build R-classes

Questions ?