Extending Adobe FrameMaker Beyond What's In the Box

- 2 ways to extend FrameMaker
  - FrameMaker Developer Kit
  - ExtendScript

- In this presentation, we would focus on ExtendScript
What is ExtendScript?

- Extended Implementation of JavaScript
- Used by many Adobe applications that provide a scripting interface

Provides:

- Development and Debugging Tools
- User Interface Development Tools
- Extensions
- Inter-application communication and messaging
Development and Debugging tools

- **ExtendScript Toolkit**: IDE for ExtendScript
  - Debugging Features:
    - Single-step through.
    - Breakpoints.
    - Watch
    - ...
  - Coding Aids
    - Code Completion
    - Code Profiling
    - Inspecting Object Models.
    - ...
User-interface development tools

- Richer UI using \textit{ScriptUI} class
- Create Modal / Floating Palettes
- Rich control set
  - Panel / TabbedPanel / Tab / Group
  - TreeView / ListView
  - ProgressBar
  - Slider
  - FlashPlayer
- Supports Localization Framework for UI
  - \texttt{msg = \{} en: "Hello, world", de: "Hallo Welt" \};
  - \texttt{alert (msg);}
User-interface development tools (examples)

- S1000D dialog shipping in FM10
Inter-application communication and messaging

- Provides a common scripting environment for all Adobe JavaScript-enabled applications
  - Adobe After Effects
  - Adobe Bridge
  - Adobe Illustrator
  - Adobe InCopy
  - Adobe InDesign
  - Adobe Indesign Server
  - Adobe Photoshop
  - Adobe Robohelp

- Startup Scripts located at:
  - C:\Program Files\Common Files\Adobe\Startup Scripts CS5
Inter-application communication and messaging

- Cross-DOM Functions
  //Opens files in Illustrator Version 12
  illustrator12.open(files);
  
  //execute a Photoshop script in the highest available version
  photoshop.executeScript (myPSScript)

- Messaging Framework
  - Send Message to other message enabled applications
  - BridgeTalk Message Object
  - Complete access to Object model of other applications
Extensions: External communication using **Socket**

- **Socket** object supports functionality to connect to remote computer over TCP/IP

- Example of HTTP Call:

  ```java
  //create Socket Object
  conn = new Socket;

  // access Adobe’s home page
  if (conn.open ("www.adobe.com:80")) {

    // send a HTTP GET request
    conn.write ("GET /index.html HTTP/1.0\n\n");

    // and read the server’s reply
    reply = conn.read(999999);
    conn.close();
  }
  ```
Extensions: File System Access using File and Folder

- Access file system using **File** and **Folder** Objects

- **File Object**
  - openDialog() / saveDialog()
  - open() / close() / read() / write() / seek() ....

- **Folder Object**
  - selectDialog()
  - create() / remove() / getFiles() / ...
  - appData / commonFiles / desktop / myDocuments / startup / userData ...
Extensions: External C/C++ code using *ExternalObject*

- Load C and C++ shared library as *ExternalObject* instance
- Extend the JavaScript DOM
- Opens up opportunities to extend functionality using C/ C++ code libraries
- Example

```javascript
//Create ExternalObject and Load the library
mylib = new ExternalObject ("lib:" + samplelib);

// access functions directly from ExternalObject instance
var a = mylib.method_abc(1,2.0,true, "this is data") ;
mylib.unload() ;
```
FM ExtendScript Support

- Built over FDK.
- Scripting support itself is implemented as a FDK Client
- FDK Client → ExtendScript
  - `F_ApiCallClient("ScriptingSupport", <filepath-to-script>)`
- ExtendScript → FDK Client
  - `CallClient("My Client", "Hello there")`

Note: ExtendScript is not replacement for FDK. It’s just another option to extend FrameMaker.
FM ExtendScript Menu

- Menu Items

![Menu Items](image)
Script Catalog

- Manage Favorites
- Scripts at FM Launch
- Scripts that listen to Notifications
FDK v/s ExtendScript

**FDK**
- Set of C Libraries
- Dev Profile
- Need to compile and create plugin that can be shared.
- Visual Studio needed
- Options to create UI

**ExtendScript**
- Scripting Language (JavaScript)
- Everyone: Easier to program and learn
- No compilation needed. Directly script can be shared
- ExtendScript IDE provided
- Richer options to create UI using ScriptUI
  - Creating modal and modeless dialogs
  - Rich set of controls
- Communicating between TCS products
  - Modifying an image in Photoshop from FM
FDK v/s ExtendScript (FM functionality examples)

**FDK**

- Import PDF Comments
- Doc Compare
- ...

**ExtendScript**

- Complete S1000D functionality
- More to come....
Type Of Scripts: Automate Tasks / Utilities

- Automate repetitive tasks
- Write utilities to create additional functionality
- Everything that can be accessed through FDK is available through ExtendScript as well
- Quick access to documents / objects
  ```javascript
  //Get Active Doc
  activeDoc = app.ActiveDoc;

  //Iterate Markers
  var m = activeDoc.FirstMarkerInDoc;
  while(m.ObjectValid()) {
    //Do Something
    m = m.NextMarkerInDoc;
  }
  ```
Type Of Scripts: Notifications Based

- Register for notifications
  
  //Register for notification
  Notification(FA_Note_PreOpenDoc,true)

- Implement Notification Handler
  
  //This will be called for registered notification
  function Notify(note,obj,sparm,iparm){
    switch (note) {
      case FA_Note_PreOpenDoc:
        // do something
          // do something
          break;
      }
  }
Type Of Scripts: Menu Commands

- Register Menu / Commands

  //Create a menu item in Table Menu
  var docMenu = app.GetNamedMenu("TableMenu")
  var sqlMenu = DefineMenu("SQLite", "SQLite");
  docMenu.AddMenuToMenu(sqlMenu);
  var sqlCmd1 = DefineCommand(1, "Resolve", "Resolve Tables", "");

- Implement Command Handler

  //This will be called when menu item is clicked
  function Command(cmd){
    switch(cmd){
      case 1:  loopAllSQLTableMarkers(); break;
      case 2:  runQueryOnSelectedTable(); break;
    }
  }
Type Of Scripts: Startup

- Register scripts to AutoRun at startup
  - Copy files / configs from central location
  - Register / Create Menu commands
  - ..... 

- Scripts in `<FMINSTALL_DIR>\startup`

- Scripts Registered through ScriptCatalog \rightarrow AutoRun