0077 Net2Html5 (aka Silver Lining)

Business Model (BM)

Document ID: 0077 Net2Html5-BM-001

Version 0.1

December 17, 2016

Table of Contents

[1 Executive Summary 5](#_Toc331009579)

[2 Business Model Diagram 6](#_Toc331009580)

[3 Roles and Responsibilities 7](#_Toc331009581)

Revision & Sign-off Sheet

**Document Properties**

| Item | Details |
| --- | --- |
| Document Title | 0077 Net2Html5 Business Model (BM) |
| Author | Ken Netherland |
| Creation Date |  |
| Last Update | 12/17/2016 |

**Change Record**

| Date | Author | Version | Change Reference |
| --- | --- | --- | --- |
| Mm/dd/yyyy |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Reviewers**

| Name | Position | Version Approved | Date |
| --- | --- | --- | --- |
|  |  |  | Mm/dd/yyyy |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Project Signatures**

By placing your signature below, you indicate your approval and acceptance of this Statement of Work to be complete and accurate portrayal of the project objectives.

|  |  |  |  |
| --- | --- | --- | --- |
| Name |  |  |  |
| Title  Division | Signature |  | Date |
| Name |  |  |  |
| Title  Division | Signature |  | Date |
| Name |  |  |  |
| Title  Division | Signature |  | Date |

# Executive Summary

The purpose of this project will be to provide a framework for converting .NET applications written in C# to TypeScript/Javascript and HTML5. The ultimate goal (later scope) is to convert the Hydra application, with the intermediate result of providing a framework that will be promoted as an open source, extensible platform for commercial use. The platform will be extensible for allowing any type of .NET application to be converted, i.e. Windows Forms, Classic WebForms, WPF applications, etc., but the initial offerings within the scope of this project will only include conversion of Silverlight/XAML and Console apps. This will include a TypeScript based engine to replace System.Console as well as the Silverlight XAML composition and rendering engine. The technology will allow continued support in C# by enabling browser debugger support for C# code (.cs files) through the use of source maps.

# Business Model Diagram



# Roles and Responsibilities

**Conversion**

**Visual Studio**

**Features:**

* Provides the ability to create Visual Studio packages to extend Visual Studio
  + Provides ability to extend the user interface using the command model and the Visual Studio Command Table (.vsct file)
  + Provides the ability to hook into build events via DTE (Visual Studio automation object model).
  + Provides ability to hook into solution events via IVsSolutionEvents, IVsSolutionLoadEvents
  + Provides ability to manage and hook into events related to the Solution Explorer hierarchy, via IVsHierarchyEvents
  + Provides ability to create Transformers to transform a source file into another related file via the IVsSingleFileGenerator
  + Provides ability to add errors to Error List Tool Window via the ErrorListProvider
  + Provides the ability to interact with the following:
    - Visual Studio Shell via IVsShell
    - Visual Studio UI via IVsUIShell
    - Hierarchies via IVsHierarchy and IVsUIHierarchy
  + Provides the ability to add the following:
    - Tool Windows
    - Custom Editors
* Provides MSBuild to read and modify Solution files, Project files, and Project items (see VisualStudioProvider)
* Provides the TypeScript framework and compiler as open-source.
  + Provides a compiler to transpile TypeScript code to Javascript
  + Provides a generator to create Source maps to map Javascript to another code file for debugging purposes

**Net2Html5 Package**

**Features:**

* Provides a framework for converting a C# project to TypeScript and Html5
  + Provides a tool bar button to convert the currently selected project(s).
  + Provides a wizard to walk the developer through the options of converting a project to Html5
    - Provides ability to create or select a Web project to host the converted application
    - Provides ability to select root scripts folder
    - Creates sub-files (TypeScript files) for each C# file in project
    - Creates sub-files (Javascript files) for each TypeScript file.
  + Updates automatically based on changes
    - Changes to C# files will result in updated TypeScript files when transformed manually by user, reopened, or rebuilt
    - Added files will be detected allowing application of transform templates.
  + Creates converted application in Web Project host when rebuilt.
    - Creates sub folders in Scripts folder for each converted project.
    - Creates a CodeUnits folder in which to copy each TypeScript file and corresponding Javascript files (for debugging purposes)
    - Converts referenced assemblies
    - Creates metadata files for debugging purposes
    - Creates a bin folder for each project to copy a binary version of the project, which encapsulates CodeUnits and metadata in PE (Portable Executable) format. This is the HydraLib format (.HydraLib)
    - Converts XAML resources to JAML (JSON based XAML)
    - Creates a debugging web page which pulls directly from Javascript files in CodeUnits folder)
    - Creates a release web page which pulls directly from HydraLib file
    - Creates an Instances root folder to represent user sessions during runtime.
    - Adds Hydra server assemblies to Web project host for runtime support.

**Net2Html5 Configuration Tool**

**Features:**

* Displays and Object graph of Sources for a configured solution
  + User can add a project source
  + User can add an assembly source
  + User can add a Code Location (directory of code files)
  + Projects that are set for conversion are added automatically
  + Referenced assemblies or projects are added if user navigates to “Reference Calls” from added sources (assemblies are excluded by default)
* Allows configuration of how the conversion process behaves
  + Allows multi-selection of the follow levels of a source
    - Projects, Assemblies, Code Locations
    - Namespaces
    - Types
      * Including the following:
        + Classes (including Attributes)
        + Structs
        + Interfaces
        + Delegates
        + Enumerations
    - Members
      * Includes the following:
        + Constructors
        + Events
        + Fields
        + Methods
        + Operators
        + Properties
        + Enumeration Items
  + Allows a conversion action to be performed on selected nodes
    - Actions are:
      * Print call graph (text file)
      * Include/exclude from conversion (nodes will be visibly marked as such)
      * Apply Stub Instructors (nodes will be visibly marked as such)
  + Allows a conversion action to be performed on a single node
    - For Assemblies/Projects
      * Redirect to a Code Location (Assembly/Project Redirector)
  + Allows applying an Extension Assembly
    - User selects assembly which has Extension (see Extensions below)
* Allows saving of the configuration
  + All nodes and applied actions will save to ConversionSettings.Net2Html5Config file (JSON) for the Host Web project
* Allows opening a saved configuration
  + User can open from Windows Explorer (standalone tool)
  + User can double click in Host Web project (opens as VS Editor)
* Shows indicators for nodes detected as changed (source nodes only)
  + Projects
    - Detects if project file or any code files within the project have changed
  + Assemblies
    - Detects if assembly has changed
  + Code Locations
    - Detects if directory layout has changed (files, folders)
    - Detects if any of the files themselves have changed
* Allows updating (recreation) of source nodes
* Allows navigating to source code for a node
  + Can navigate to original code
  + Can navigate to TypeScript generated code
  + Can navigate to JavaScript generated code
* Shows indicators for nodes that have errors
  + Detects if a project or assembly has TypeScriptCompile errors and indicates as such

Notes:

For the last four features, will require configuration to contain the following:

* For Assemblies
  + Assembly file path
  + Assembly file hash
* For Projects
  + Project file path
  + Project file hash
* For Code locations
  + Code location directory
  + Code layout (hierarchy) JSON hash
    - Includes in hierarchical fashion
      * Directory names
      * Code file paths, hashes for each file
* For Other Nodes
  + Input (C#) source code line number, column number
  + TypeScript output file path, file hash (only top levels)
  + TypeScript output source code line number, column number
  + Input source AST node path identifier
  + TypeScript source AST node path identifier
  + JavaScript source AST node path identifier
* Provides a library-based (non-UI) and in-memory representation for conversion process use.
* Nodes get created by conversion process if do not exist in ConversionSettings.Net2Html5Config
* Warns user if nodes do not exist

**Conversion Developer**

**Responsibilities**

* Installs Net2Html5 package
* Converts projects within a solution.
  + Does one of the following:
    - Selects a project and clicks ‘Convert [Project]’ from Solution Explorer tool bar.
    - Right clicks a project and selects menu ‘Convert to HTML5’
  + Walks through wizard and selects options
* Builds Web project
* Selects one of the web pages to run converted application.
* Has ability to “freeze” a typescript file to prevent it from being copied to CodeUnits folder or bin folder. This allows the developer to continue development of the TypeScript code without changes being overwritten by conversion.

**Extensions Developer**

**Responsibilities**

* Develops Pre-Conversion Extensions
  + Develops CSharpParser Code Actions
* Develops TypeScript Output Extensions
  + Configures Stub Instructors
  + Configures Assembly/Project Redirectors
  + Develops CsToTs Transforms
  + Develops Type Wrappers
  + Develops Type Member Handlers
  + Develops Language Feature Handler
  + Develops Node Handlers
* Develops Javascript Output Extensions
  + Develops Source Map Handlers
  + Develops Node Handlers
* Develops Redirector Extensions
  + Configures Assembly Input Redirectors
  + Configures CodeUnit Output Redirectors

**Usage**

**Hydra Presentation Framework**

**Features:**

* Provides a WPF-like Runtime Environment
* Provides a rich user interface based on Html5
* Provides standard WPF-like controls with ability to extend
* Allows code to be maintained and debugged in .NET/C#
* Allows for code to be wrapped in .NET-like assemblies
* Provides many of the same code libraries as the Silverlight Runtime
* Provides a Console Runtime Environment (loads Console as separate browser window)

**ReactJs**

<https://facebook.github.io/react/>

**Features:**

* Provides a Javascript supported component-based UI framework
  + UI’s are built completely from components that render HTML
  + HTML is added declaratively to components, eliminating the need to interact directly with the browser DOM.
  + Closely mirrors the rendering model of XAML
  + Supports two-way binding using “this.props” to mirror WPF binding
* Supported by TypeScript compiler
  + Compiler accepts JSX/TSX files which are Javascript/TypeScript files that include declarative markup
  + To support UI components that can continue to inherit from FrameworkElement and UIElement, mixins can be used to apply React components similar to FrameworkTemplates (ControlTemplate)

**User**

**Responsibilities**

* Navigates to a Hydra Presentation Framework application URL
* Downloads bootstrapper (script dynamically created by server)
* Bootstrapper downloads Fusion API
* Bootstrapper uses Fusion API to download Hydra Presentation Runtime Environment
* Runtime Environment looks for DIV data sources (can be more than one)
* Downloads and launches startup Assembly from data source

**Runtime Server Extensions Developer**

**Responsibilities**

* Creates extension to wrap and intercept server calls by the application (i.e. Domain services)
* Creates obfuscation service extensions
* Creates encryption service extensions
* Creates compression service extensions
* Creates Runtime Environment extensions (can override base methods)