New MP Command
"Load HTML Form in Edge Browser"

Table of Contents

1 “Load HTML Form”

1.1 *Reasons for the Upgrade*

The existing MP command “Load HTML Form” allows to retrieve user input from a prompted custom HTML file and save data to a DataShare file. The problem is that the MP command opens the HTML form in Internet Explorer browser which is already outdated (MS announced IE end of life in 07.2022) and can cause formatting frustration when it gets jumbled when presented in SA.

It is required to update the current IE web browser to Chromium-based web browser for use of new web technologies (HTML, CSS, and JavaScript)

1.2 *Implementation*

1.2.1 Microsoft Edge Browser

Added support of Chromium-based Microsoft Edge browser to the application using MS Edge WebView2 control: https://learn.microsoft.com/en-us/microsoft-edge/webview2/

The Microsoft Edge WebView2 control allows to embed web technologies (HTML, CSS, and JavaScript) in the native apps. The WebView2 control uses Microsoft Edge as the rendering engine to display the web content in native apps.

The Edge browser also provides ability to auto validate client-side form.

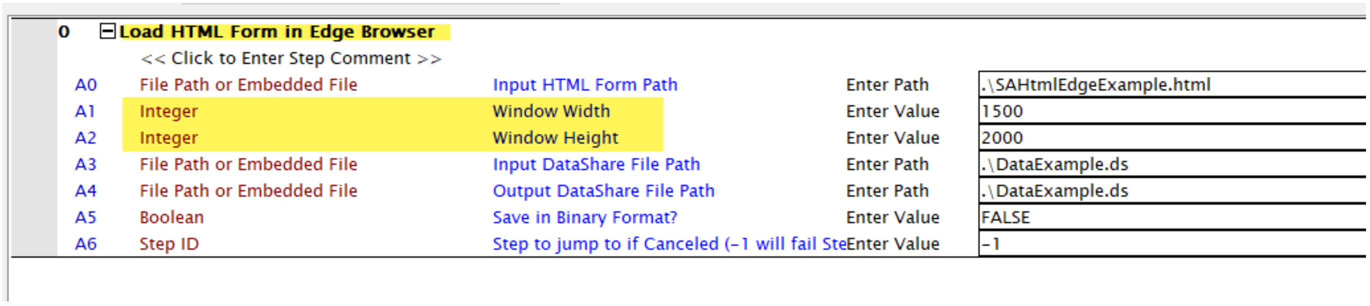
1.2.2 MP Commands Working with Web Browsers

Now, SA supports two MP commands that allow collecting user input from HTML forms:

1. Load HTML Form
2. Load HTML Form in Edge Browser

The original **“Load HTML Form”** MP command functionally remained the same. It was modified slightly to allow user to control the size of the “browser” window called by SA. Added two MP arguments – window width and height. Now, user can alternate between large and small forms in the same MP script.

To satisfy the requirement, a new MP command **“Load HTML Form in Edge Web Browser”** has been introduced to the application under “File Operations/DataShare operations” category.



0 Load HTML Form in Edge Browser

<< Click to Enter Step Comment >>

A0	File Path or Embedded File	Input HTML Form Path	Enter Path	.\SAHtmlEdgeExample.html
A1	Integer	Window Width	Enter Value	1500
A2	Integer	Window Height	Enter Value	2000
A3	File Path or Embedded File	Input DataShare File Path	Enter Path	.\DataExample.ds
A4	File Path or Embedded File	Output DataShare File Path	Enter Path	.\DataExample.ds
A5	Boolean	Save in Binary Format?	Enter Value	FALSE
A6	Step ID	Step to jump to if Canceled (-1 will fail Ste	Enter Value	-1

#	Argument Type	Argument Name	Description

A0	File Path or Embedded File	Input HTML Form Path	The HTML custom form file location. If file doesn't exist, the command fails.
A1	Integer	Window Width	Defines window width size in pixels.
A2	Integer	Window Height	Defines window height size in pixels
A3	File Path or Embedded File	Input DataShare File Path	The input DataShare file location. The file contains initial values (default values) for HTML form. If file exists and has data, then the application tries to restore data from DataShare file to HTML form presenting values from the file.
A4	File Path or Embedded File	Output DataShare File Path	The output DataShare file location where the extracted from HTML form user data will be saved. If DataShare file doesn't exist, new file is automatically created. New arguments are automatically appended to the existing DataShare file on Save operation.
A5	Boolean	Save in Binary Format?	The argument defines file format for DataShare output file.
A6	Step ID	Step to jump to if Canceled (-1 will fail Step on Cancel)	The argument defines a MP step in case of operation has been cancelled.

Note: Input and Output DataShare File paths may be the same.

1.2.3 "Edge Browser – User Input for DataShare file" Dialog

In runtime, the application presents the specified (see A0) custom HTML form in the SA dialog "Edge Browser – User Input for DataShare file". The window size is controlled by the command. User can fill in the form.

1.2.3.1 Save and Cancel Operation

The form should provide functionality of two buttons – "Save" and "Cancel". For example,

```
<!-- ---------------------- -->
<!-- **** Buttons Test **** -->
<!-- ---------------------- -->
<br /> <br />
<button class="buttonSave" type="button" onclick="SendSubmitMessage()">Save</button>
<button class="buttonCancel" type="button" onclick="SendCancelMessage()">Cancel</button>
```

The HTML form and SA are exchanging web messages. The SA application can execute javascript.

Currently, the following javascript functions should be added to the form to satisfy "saving requirement:

- function **GetFormInputs()** (this function name is reserved and expected)

  This function sends one-by-one each input/select form field in the following CSV format:

  <tag>,<type>,<id>,<name>,<value>,<checked>
- function SendSubmitMessage()

This function

- collects all user inputs from the form.
- validates them.
- if valid, sends collected inputs to SA.
- closes the form.

SA is expecting to receive the following keywords to start/end the saving process:

- 'SaveData: Start'
- 'SaveData: End'

Here is a javascript function example:

```
function SendSubmitMessage() {
        // validate user input
        if (!document.getElementById('SAForm').checkValidity()) {
                alert("Please enter values to the required fields");
                return;
        }

        // indicate to SA that the saving process starts
        window.chrome.webview.postMessage('SaveData: Start');

        // send all inputs in the predefined CSV format
        GetFormInputs();

        // indicate to SA that the saving process ends
        window.chrome.webview.postMessage('SaveData: End');
        }
```

- function SendCancelMessage()

  SA is expecting to receive **Cancel** keyword from the form.
  Here is a javascript function example:

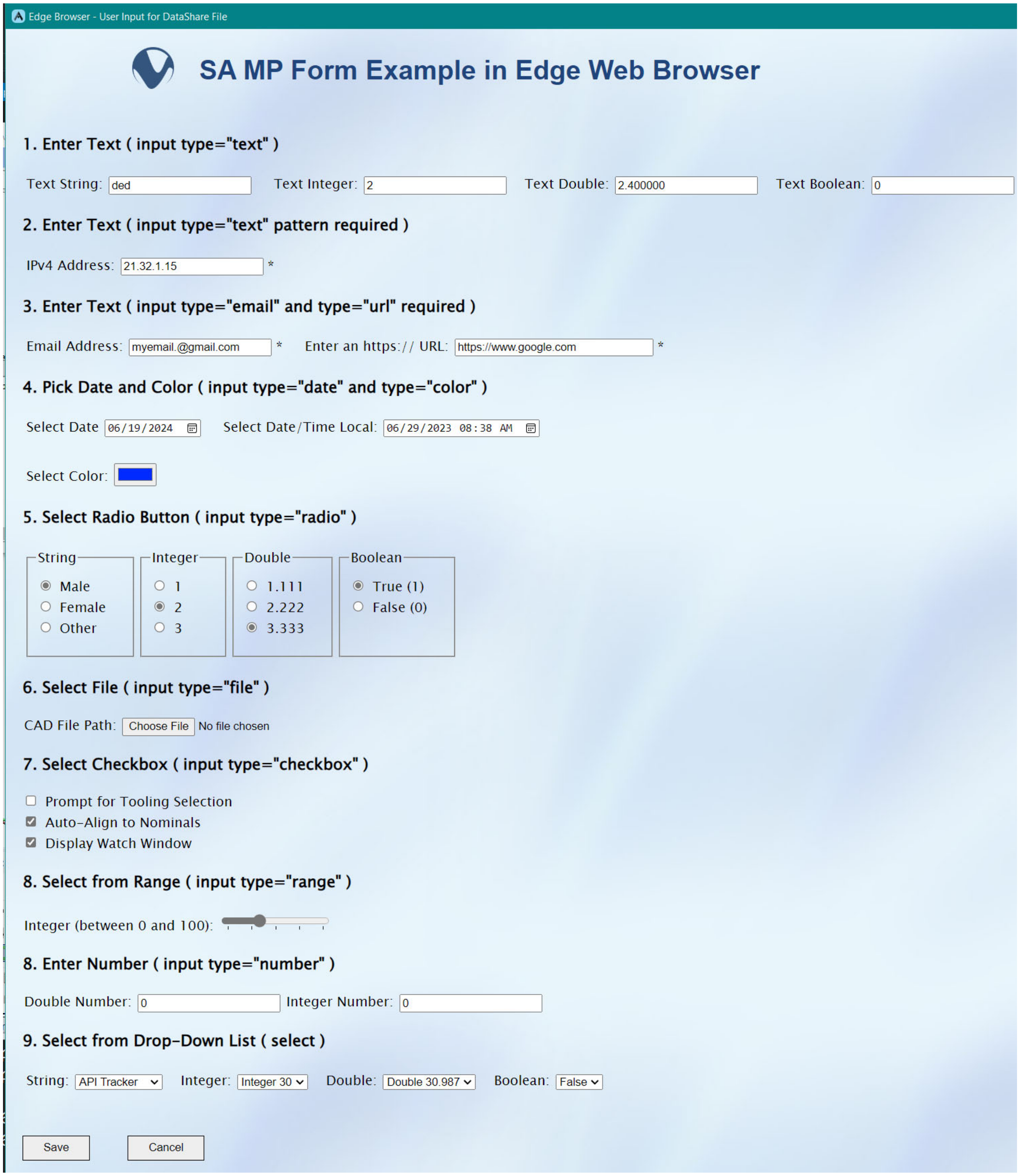
```
function SendCancelMessage() {
        window.chrome.webview.postMessage('Cancel');
                }
```

On “Cancel” the MP step fails or jumps to another step (see Arguments Table above).

On “Save” the data are automatically saves in the DataShare file of specified type (binary or ASCII). Once DataShare file has been saved, it can be loaded by “Load DataShare File” MP command and used with the rest of the DataShare MP commands to handle data.

1.2.4 HTML Form Rules

Here is an example of possible HTML form:



Edge Browser - User Input for DataShare File

SA MP Form Example in Edge Web Browser

1. Enter Text (input type="text")

Text String: ded Text Integer: 2 Text Double: 2.400000 Text Boolean: 0

2. Enter Text (input type="text" pattern required)

IPv4 Address: 21.32.1.15 *

3. Enter Text (input type="email" and type="url" required)

Email Address: myemail.@gmail.com * Enter an https:// URL: https://www.google.com *

4. Pick Date and Color (input type="date" and type="color")

Select Date 06/19/2024 Select Date/Time Local: 06/29/2023 08:38 AM

Select Color: 🟦

5. Select Radio Button (input type="radio")

String: [✓ Male] Female Other

Integer: 1 [✓ 2] 3

Double: 1.111 2.222 [✓ 3.333]

Boolean: [✓ True (1)] False (0)

6. Select File (input type="file")

CAD File Path: Choose File No file chosen

7. Select Checkbox (input type="checkbox")

- Prompt for Tooling Selection
- [✓ Auto-Align to Nominals]
- [✓ Display Watch Window]

8. Select from Range (input type="range")

Integer (between 0 and 100):

8. Enter Number (input type="number")

Double Number: 0 Integer Number: 0

9. Select from Drop-Down List (select)

String: API Tracker Integer: Integer 30 Double: Double 30.987 Boolean: False

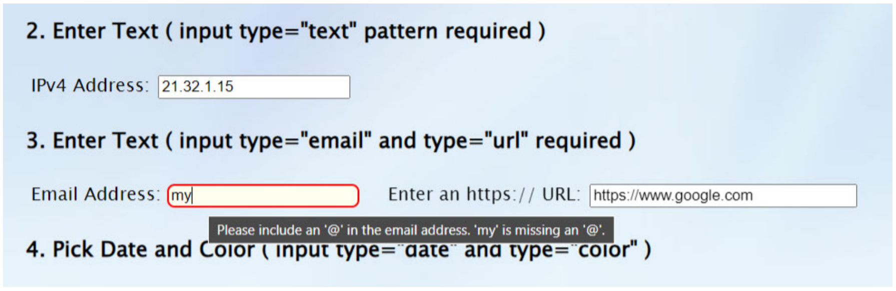
Save Cancel

1.2.4.1 Client-Side Form Validation

The Edge Brower has an ability to validate most of the user data without relying on JavaScript. The validation is based on the following element attributes:

- ✓ required
- ✓ minlength and maxlength
- ✓ min and max
- ✓ type
- ✓ pattern
- ✓ regular expression

For example, the email field is marked as required and the help balloon is popping up automatically on invalid input.



2. Enter Text (input type="text" pattern required)

IPv4 Address: 21.32.1.15

3. Enter Text (input type="email" and type="url" required)

Email Address: my Enter an https:// URL: https://www.google.com

Please include an '@' in the email address. 'my' is missing an '@'.

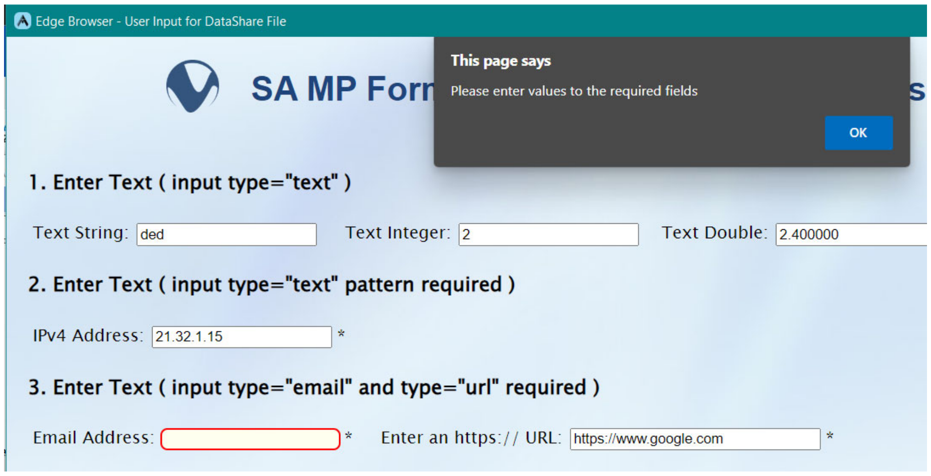
4. Pick Date and Color (input type="date" and type="color")

To use of client-side form auto validation and JavaScript, it is important to

- ✓ add “id” to the form: `<form id="SAForm">`
- ✓ Use the form’s checkValidity() function on Save button click:

```
// validate user input
if (!document.getElementById('SAForm').checkValidity()) {
      alert("Please enter values to the required fields");
      return;
}
```

In case of the invalid input, the saving operation doesn’t start till form satisfies the requirements:



Edge Browser - User Input for DataShare File

SA MP Forn ... s

This page says

Please enter values to the required fields

OK

1. Enter Text (input type="text")

Text String: ded Text Integer: 2 Text Double: 2.400000

2. Enter Text (input type="text" pattern required)

IPv4 Address: 21.32.1.15 *

3. Enter Text (input type="email" and type="url" required)

Email Address: * Enter an https:// URL: https://www.google.com *

Although HTML form can be created per customer custom as it can be, the following rules should be followed when writing HTML code to ensure proper parsing and restoring operations to/from DataShare.

1.2.4.2 Output Argument Name

The argument name should be specified by HTML ***name*** attribute.

1.2.4.3 Output Argument Type

Currently, the application supports the following data types extracted from HTML form:

- String (S)
- Integer (I)
- Double (D)
- Boolean (B)

In order to identify each value the HTML ***id*** attribute must be specified. The id attribute can be unique as needed. In order to identify type of entered value, the application uses first character of HTML ***id*** attribute.

For example, id="S123" or id="S87654" will be identified as Strings. The examples below illustrate HTML <input> and <select> tags use.

1.2.4.4 HTML <input> Tag

1.2.4.4.1 Input "text" Type (Text Field)

The "text" type value can represent String, Integer, Double, and Boolen value types.

```
<h3>1. Enter Text ( input type="text" )</h3>
<table>
       <tbody>
       <td>
              <label for="S1">Text String:</label>
              <input type="text" id="S1" name="String" value="" />
       </td>

       <td>
              &emsp;<label for="I1"> Text Integer:</label>
              <input type="text" id="I1" name="Integer" value="" />
       </td>

       <td>
              &emsp;<label for="D1">Text Double:</label>
              <input type="text" id="D1" name="Double" value="" />
       </td>

       <td>
              &emsp;<label for="B1">Text Boolean:</label>
              <input type="text" id="B1" name="Boolean" value="" />
       </td>
       </tbody>
</table>
```

1.2.4.4.2 Input “text” Type with Pattern (Text Field)

```
<!-- -------------------------- -->
<!-- **** Input IP address **** -->
<!-- -------------------------- -->
    <h3>2. Enter Text ( input type="text" pattern required )</h3>
    <table>
           <tbody>
           <td>
                  <label for="S57">IPv4 Address:</label>
                  <input align="left" type="text" id="S57" name="IP" pattern="(?:(?:25[0-4]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-4]|2[0-4][0-9]|[01]?[0-9][0-9]?)" required placeholder="123.1.12.123">&nbsp;*
                  <span></span>
           </td>
           </tbody>
    </table>
```

1.2.4.4.3 Input “number” Type (Text Field)

<input type="number" id = "D1" name="Circle Diameter" value=0>

<input type="number" id = "I1" name="Desired Measurement Count" value=0>

1.2.4.4.4 Input “email” and “url” types (Text Fields)

```
    <!-- ----------------------------- -->
    <!-- **** Input email ****         -->
    <!-- ----------------------------- -->
    <h3>3. Enter Text ( input type="email" and type="url" required )</h3>
    <table>
           <tbody>
           <td>
                  <label for="S4">Email Address: </label>
                  <input type="email" id="S4" name="email" value="" required />&nbsp;*
                  <span></span>
           </td>
           <td>
                  &emsp;
                  <label for="S41">Enter an https:// URL:</label>

                  <input type="url" name="url" id="S41"
                            placeholder="https://example.com"
                            pattern="https://.*" size="30"
                            required>&nbsp;*
           </td>
           </tbody>
    </table>
```

1.2.4.4.5 Input “radio” Type (Radio Buttons)

It is important for radio buttons of the same group to have identical name attributes, data type, and unique ids.

```
    <!-- ----------------------------- -->
    <!-- **** Radio Button Test ****   -->
    <!-- ----------------------------- -->
    <h3>5. Select Radio Button ( input type="radio" )</h3>
    <table>
           <tbody>
```

```
                <td>
                        <fieldset>
                        <legend>String</legend>
                                <input align="left" type="radio" id="S6" name="Radio String" value="male"
checked />
                                <label for="S6">Male</label><br>
                                <input align="left" type="radio" id="S61" name="Radio String"
value="female" />
                                <label for="S61">Female</label><br>
                                <input align="left" type="radio" id="S62" name="Radio String" value="other"
/>
                                <label for="S62">Other</label><br>
                        </fieldset>
                </td>
                <td>
                        <fieldset>
                                <legend>Integer</legend>
                                <input align="left" type="radio" id="I2" name="Radio Int" value="1" checked
/>
                                <label for="I2">1</label><br>
                                <input align="left" type="radio" id="I21" name="Radio Int" value="2" />
                                <label for="I21">2</label><br>
                                <input align="left" type="radio" id="I22" name="Radio Int" value="3" />
                                <label for="I22">3</label><br>
                        </fieldset>
                </td>
                <td>
                        <fieldset>
                                <legend>Double</legend>
                                <input align="left" type="radio" id="D2" name="Radio Double" value="1.111"
checked />
                                <label for="D2">1.111</label><br>
                                <input align="left" type="radio" id="D21" name="Radio Double" value="2.222"
/>
                                <label for="D21">2.222</label><br>
                                <input align="left" type="radio" id="D22" name="Radio Double" value="3.333"
/>
                                <label for="D22">3.333</label><br>
                        </fieldset>
                </td>
                <td>
                        <fieldset>
                                <legend>Boolean</legend>
                                <input align="left" type="radio" id="B2" name="Radio Boolean" value="1"
checked />
                                <label for="B2">True (1)</label><br>
                                <input align="left" type="radio" id="B21" name="Radio Boolean" value="0" />
                                <label for="B21">False (0)</label><br>
                                <br />
                        </fieldset>
                </td>
                </tbody>
        </table>
```

1.2.4.4.6 Input “file” Type (File Browser)

The selected filename path should have String identification as in example below:

<input type="file" size = "50" id="S4" name="CAD File Path" value="">

Note: Before presenting HTML form to user, the application restores previously saved values from DataShare file. An input field of “file” type has read only “value” attribute for security purpose. The application can’t populate (write) pre-stored filename to reopened HTML form. The re-opened form filename path initial value is always empty. User doesn’t have to re-enter filename again on the re-opened form; unless he wants to change the path.

```
<!-- ----------------------------- -->
<!-- **** Input file Test ****     -->
<!-- ----------------------------- -->
<h3>6. Select File ( input type="file" )</h3>
<label for="S7">CAD File Path:</label>
<input type="file" id="S7" name="CAD File Path" value="">
```

1.2.4.4.7 Input "checkbox" Type (Checkboxes)

The HTML checkboxes represent Boolean values. See example below for proper type/name identification:

```
<!-- ----------------------------- -->
<!-- **** Input checkbox Test **** -->
<!-- ----------------------------- -->
<h3>7. Select Checkbox ( input type="checkbox" )</h3>

<input type="checkbox" id="B4" name="Prompt for Tooling Selection" value="" />
<label for="B4">Prompt for Tooling Selection</label><br>

<input type="checkbox" id="B5" name="Auto-Align to Nominals" value="" />
<label for="B5">Auto-Align to Nominals</label><br>

<input type="checkbox" id="B6" name="Display Watch Window" value="" />
<label for="B6">Display Watch Window</label><br>
```

1.2.4.4.8 Input "date" and "color" Types

```
<!-- ----------------------------- -->
<!-- **** Input date Test ****     -->
<!-- ----------------------------- -->
<h3>4. Pick Date and Color ( input type="date" and type="color" )</h3>
<table>
       <tbody>
       <td>
              <label for="S5">Select Date</label>
              <input type="date" id="S5" name="Picked Date" />
       </td>

       <td>
              &emsp;
              <label for="S51">Select Date/Time Local:</label>
              <input type="datetime-local" id="S51" name="Local Time">
       </td>
       </tbody>
</table>
<br />
<table>
       <tbody>
       <td>
              <label for="S52">Select Color:</label>
              <input type="color" id="S52" name="Picked Color" value="#ff0000">
       </td>
       </tbody>
</table>
```

1.2.4.4.9 Input “range” Type

The HTML “range” type represent a number value. Here is an example of selecting an integer value.

```
              <!-- ---------------------------- -->
              <!-- **** Input range Test ****     -->
              <!-- ---------------------------- -->
              <h3>9. Select from Range ( input type="range" )</h3>
              <label for="vol">Integer (between 0 and 100):</label>
              <input type="range" id="I4" name="Integer from Range" min="0" max="100"
list="markers">
              <datalist id="markers">
                     <option value="0"></option>
                     <option value="25"></option>
                     <option value="50"></option>
                     <option value="75"></option>
                     <option value="100"></option>
              </datalist>
```

1.2.4.5 HTML <select> Tag (Drop-Down List)

The type/name identification should be added to <select> tag.

```
              <!-- ---------------------------- -->
              <!-- **** drop-down list Test **** -->
              <!-- ---------------------------- -->
              <h3>8. Select from Drop-Down List ( select )</h3>
              <table>
                     <tbody>
                     <td>
                            <label for="S8">String:</label>
                            <select id="S8" name="Drop Down String" form="SAForm">
                                   <option value="Faro">Faro Tracker</option>
                                   <option selected value="Leica">Lieca Tracker</option>
                                   <option value="API">API Tracker</option>
                            </select>
                     </td>

                     <td>
                            &emsp;<label for="I3">Integer:</label>
                            <select id="I3" name="Drop Down Integer" form="SAForm">
                                   <option selected value="10">Integer 10</option>
                                   <option value="20">Integer 20</option>
                                   <option value="30">Integer 30</option>
                            </select>
                     </td>

                     <td>
                            &emsp;<label for="D3">Double:</label>
                            <select id="D3" name="Drop Down Double" form="SAForm">
                                   <option selected value="1">Double 10.346</option>
                                   <option value="20.543">Double 20.543</option>
                                   <option value="30.987">Double 30.987</option>
                            </select>
                     </td>

                     <td>
                            &emsp;<label for="B3">Boolean:</label>
                            <select id="B3" name="Drop Down Boolean" form="SAForm">
```

```
                    <option selected value="1">True</option>
                    <option value="0">False</option>
                </select>
            </td>
            </tbody>
        </table>
```