surveyman.js Documentation

Release 1.5

Emma Tosch

September 07, 2014

The surveyman.js repository contains three modules. Together these three modules form the SurveyMan runtime system, contained in the SurveyMan namespace. This runtime system receives a JSON representation of a survey and executes that survey (with the help of a human!) in a browser. These three modules are meant to be used with the SurveyMan server and static analyzer.

Currently this code expects to be executed inside http://github.com/SurveyMan/SurveyMan/src/main/resources/HTMLSkeleton.html.

CHAPTER 1

Install

To install locally, run

```
npm install surveyman
```

CHAPTER 2

Dependencies

- underscore
- jQuery
- seedrandom

If you use these modules with the SurveyMan Java backend, everything is good to go! If you're using your own backend or otherwise modifying some part of the pipeline, you will need to ensure the following are present in your HTML for everything to work:

First, make sure you have the following in the head:

```
<script type="text/javascript" src="http://surveyman.github.io/surveyman.js/survey.js"></script>
<script type="text/javascript" src="http://surveyman.github.io/surveyman.js/interpreter.js"></script
<script type="text/javascript" src="http://surveyman.github.io/surveyman.js/display.js"></script>
```

If you are using AMT as your backend, you will also need a link to the submission script in the head, per the AMT documentations. If you are using a local backend, you will need some way to capture the assignment id, since it's used to seed the random number generator. The SurveyMan backend generates the following when it is being run locally:

```
<script type="text/javascript">
   $.ajaxSetup({async:false});
   var turkSetAssignmentID = function () {
        $.get("assignmentId", function(_aid) {
            console.log("Just pulled assignment Id : " + _aid);
            document.getElementById("assignmentId").value = _aid.trim();
            aid = _aid;
          });
       };
</script>
```

turkSetAssignmentId is an AMT-defined function. Since *SurveyMan.display.ready* expects it, we define a local version here. AMT also injects an *assignmentId* element, so when we run locally, we add an element with this id to our form.

SurveyMan generates a form to be sent with a POST; although a user-defined version could simply collect data in a Javascript object and POST this back to a local server, we wanted to be able to write one version to work with both AMT and a local server.

At the end of the body, SurveyMan adds the following snippet:

```
<script type='text/javascript'>
    turkSetAssignmentID();
    var loadPreview=function(){<PROBABLY_A_CONSENT_FORM>},
        jsonizedSurvey=<JSONIZED_SURVEY>;
    Surveyman.display.ready(jsonizedSurvey, loadPreview);
</script>
```

surveyman.js

surveyman.js contains the internal Javascript representation of a survey. It is simlar to the Java representation contained in http://github.com/SurveyMan/SurveyMan. The internal representation is not visible to the user.

`SurveyMan.survey.`**`init`**`(`*jsonSurvey*`)`

Arguments

- **jsonSurvey** – A JSON representation of the survey, as defined by the JSON schema http://github.com/SurveyMan/SurveyMan/src/main/resources/schemata

Returns A Javascript Survey object

`SurveyMan.survey.`**`getOptionById`**`(`*oid*`)`

Arguments

- **oid** (*string*) – String representation of an option id

Returns A Javascript Survey Option object

`SurveyMan.survey.`**`getQuestionById`**`(`*quid*`)`

Arguments

- **quid** (*string*) – String representation of a question id

Returns A Javascript Question object

`SurveyMan.survey.`**`getBlockById`**`(`*bid*`)`

Arguments

- **bid** (*string*) – String representation of a Block id

Returns A Javascript Block object

CHAPTER 4

interpreter.js

interpreter.js executes the survey. It is initialized with a Survey object and updates the state of the interpreter in response to requests for more questions.

Under the current implmentation, the interpreter does not contain the answer set; this is held in the HTML.

`SurveyMan.interpreter.`**`init`**`(`*jsonSurvey*`)`
Initilizes survey and the interpreter's stacks.

Arguments

- **jsonSurvey** – A JSON representation of the survey, as defined by the JSON schema http://github.com/SurveyMan/SurveyMan/src/main/resources/schemata

`SurveyMan.interpreter.`**`isQuestionStackEmpty`**`()`

Returns boolean

`SurveyMan.interpreter.`**`isBlockStackEmpty`**`()`

Returns boolean

`SurveyMan.interpreter.`**`nextBlock`**`()`

Returns A Block object

`SurveyMan.interpreter.`**`nextQuestion`**`()`

Returns A Question object

`SurveyMan.interpreter.`**`handleBranching`**`(`*q*, *o*`)`

Arguments

- **q** – A Question object
- **o** – An Option object

Returns The next Question object

`SurveyMan.interpreter.`**`nextSequential`**`()`

Returns The next Question object

CHAPTER 5

display.js

This is the module that controls all of the display logic. The visible methods can be overridden in *~/surveyman/custom.js* to do things like inject timing information.

** Button Functionality**

“Next” buttons will start out hidden. When the respondent selects an answer, they appear. For a single question, its “Next” button can only be hidden again if the question is a checkbox question and the respondent unchecks all boxes.

“Submit Early” will appear alongside “Next” if the survey permits breakoff. When the respondent answers the final question, only a button reading “Submit” will appear.

showNextButton and *getOptionHTML* call *showSubmit*. We include *showSubmit* here in case the survey author wishes to override some behavior.

`SurveyMan.display.`**`hideNextButton`**`(`*q*`)`

Arguments

- **q** – A Question object

`SurveyMan.display.`**`showNextButton`**`(`*pid*, *q*, *o*`)`

Arguments

- **pid** (*string*) – The id of the div containing this question. Answered questions are pushed onto a hidden HTML “stack”.
- **q** – The Question object that’s just been answered.
- **o** – The Option object that is the answer to that question.

`SurveyMan.display.`**`showSubmit`**`(`*q*, *o*`)`

Arguments

- **q** – The current Question object.
- **o** – The answered Option object.

`SurveyMan.display.`**`getNextQuestion`**`(`*q*, *o*`)`

Arguments

- **q** – The Question just answered
- **o** – The Option that is the answer

Returns The next Question object

`SurveyMan.display.`**`showQuestion`**`(`*q*`)`

Arguments

- **q** – The Question object being shown.

`SurveyMan.display.`**`getOptionHTML`**`(`*q*`)`

Arguments

- **q** – The Question object being shown.

Returns Javascript representation of HTML elements containing Option data. If the Option data is a display, then the "Next" button is shown.

`SurveyMan.display.`**`showOptions`**`(`*q*`)`

Arguments

- **q** – The Question object whose options are being shown.

`SurveyMan.display.`**`registerAnswerAndShowNextQuestion`**`(`*pid*, *q*, *o*`)`

Arguments

- **pid** (*string*) – The id of the div containing this question. Answered questions are pushed onto a hidden HTML "stack".
- **q** – The Question object that's just been answered.
- **o** – The Option object that is the answer to that question.

`SurveyMan.display.`**`showBreakoffNotice`**`()`

Displays a predefined breakoffo notice. This can be changed in *~/surveyman/custom.js*.

`SurveyMan.display.`**`showFirstQuestion`**`()`

Displays the first question. Is called first if there is not breakoff.

`SurveyMan.display.`**`ready`**`(`*jsonizedSurvey*, *customInit*`)`

Arguments

- **jsonSurvey** – A JSON representation of the survey, as defined by the JSON schema http://github.com/SurveyMan/SurveyMan/src/main/resources/schemata
- **customInit** – The custom Javascript contained in *~/surveyman/custom.js*. This will be included in the HTML if the SurveyMan Java backend is used.

CHAPTER 6

Indices and tables

- *genindex*
- *search*