Conversational Interface Design Spec.

Version: 0.1 (draft)

# Overview

This document describes the design of the Simple Task Automation Bot and the goal is to make it easier to communicate the user experience and flows to both users and developers.

# Main dialog structure:

# Scorables:

Scorables are “catch all” rules: Regardless of which dialog is handling the conversation, the user input is always checked against these Scorables and if one of these rules is triggered, the context of the conversation may change. For example, the bot may be asking the user a “yes/no” question, but the user may instead reply “help”. No scorables are used in this bot.

# Dialogs Specification:

# RootDialog

|  |  |  |  |
| --- | --- | --- | --- |
| **Event** | **Description** | | **Notes** |
| **On Start:** | Dialog initiates with the following message: | | On Start is what initially happens when the dialog is invoked |
| **User Input:** | User will be allowed to click at one of the buttons above (or type one of those options on channels that don’t support buttons): | |  |
| **Action** | **Effect** |
| “Change Password” | Flow not implemented. Send a message to the user |
| “Reset Password” | Call to ResetPassword dialog |
| **Invalid user input:** | In case of an invalid user input: | | Assuming the user types something else and no scorable picks that up, we will reject the input and retry the question. |
| **Exit Criteria:** | This is the root dialog and therefore it never exits. | | Root dialogs shouldn’t exit. They are the very entry point where the whole experience starts. |

# ResetPassword dialog

|  |  |  |
| --- | --- | --- |
| **Event** | **Description** | **Notes** |
| **On Start:** | Dialog initiates calling the PromptStringRegex dialog to ask the user for a phone number.  Once that dialog completes, if the phone number was provided, it will be shown:    And then the PromptDate dialog will be called to ask the user for the date of birth. |  |
| **User Input:** | No user input is provided in this dialog |  |
| **Invalid user input:** | No user input is provided in this dialog |  |
| **Exit Criteria:** | If the phone number and the date of birth are correctly provided, the dialog will provide a new password and ends with a “true” value to indicate success.    If the phone number or date of birth are not correctly provided or any of those prompts is cancelled, the dialog will end with a “false” value to indicate that it failed. RootDialog will detect that and will show: |  |

# PromptStringRegex dialog

|  |  |  |  |
| --- | --- | --- | --- |
| **Event** | **Description** | | **Notes** |
| **On Start:** | Dialog initiates with the following question: | | Since the options are too many, we will use free text input here. A custom PromptString dialog with Regex validation is being used |
| **User Input:** | User will be allowed to type a phone number | | A regular expression is being used to validate the input |
| **Action** | **Effect** |
| Any input | Run a phone number regex validation |
| **Invalid user input:** | In case on an invalid phone number: | |  |
| **Exit Criteria:** | To be defined in the alternative flows below | |  |

**Alternative Flow 1**

|  |  |  |
| --- | --- | --- |
| **Condition** | An invalid phone was provided many times |  |
| **Effect** | PromptStringRegex dialog ends with the following message: |  |
| **Exit Criteria:** | PromptStringRegex throws a TooManyAttempsException | No phone number is provided |

**Alternative Flow 2**

|  |  |  |
| --- | --- | --- |
| **Condition** | A valid phone was provided |  |
|  |  |  |
| **Exit Criteria:** | A valid phone number | We have one valid phone number |

**Alternative Flow 3**

|  |  |  |
| --- | --- | --- |
| **Condition** | “Cancel” message was sent |  |
|  |  |  |
| **Exit Criteria:** | A null value | No phone number is provided |

# PromptDate dialog

|  |  |  |  |
| --- | --- | --- | --- |
| **Event** | **Description** | | **Notes** |
| **On Start:** | Dialog initiates with the following question: | | Since the options are too many, we will use free text input here. A custom Prompt<T,U> dialog with custom validation is being used |
| **User Input:** | User will be allowed to type a date | | DateTime.TryParseExact is being used to determine if the date is a valid one |
| **Action** | **Effect** |
| Any input | Run a basic date validation (format) |
| **Invalid user input:** | In case on an invalid date: | |  |
| **Exit Criteria:** | To be defined in the alternative flows below | |  |

**Alternative Flow 1**

|  |  |  |
| --- | --- | --- |
| **Condition** | An date was provided many times |  |
| **Effect** | PromptDate dialog ends with the following message: |  |
| **Exit Criteria:** | PromptDate throws a TooManyAttempsException | No date is provided. DateTime.MinValue is returned |

**Alternative Flow 2**

|  |  |  |
| --- | --- | --- |
| **Condition** | A valid date was provided |  |
|  |  |  |
| **Exit Criteria:** | A valid date | We have one valid date |

**Alternative Flow 3**

|  |  |  |
| --- | --- | --- |
| **Condition** | “Cancel” message was sent |  |
|  |  |  |
| **Exit Criteria:** | A DateTime.MinValue date | No date is provided. DateTime.MinValue is returned |