GEKI

Advanced User Interfaces Course Repository

Configuration

Action Files:

Action Files MUST contain an array of actions configured as follows:

Remember that the order matters!

```
[
    {
        "parseIdentifier": "START",
        "attributes": {}
    },
    {
        "parseIdentifier": "END",
        "attributes": {}
    },
]
```

Actions:

Actions are composed as follows:

```
{
    "parseIdentifier": "START",
    "attributes": {}
}
```

Other two *OPTIONAL* attributes can be specified:

```
{
    "parseIdentifier": "START",
    "attributes": {},
    "identifier": "__ID__",
    "timeout": 10
}
```

`identifier` m if specified, sets an identifier for that action so that can be used to the Skip Action.
`timeout` if specified (in seconds), kills the action after a certain amount of time.

StartAction

An action that identifies the game's start point.
Start and End actions should be used ONLY in the mail loadable action file!

```
{
    "parseIdentifier": "START",
    "attributes": {}
```

```
}
```

Available Attributes

No attributes are needed.

End Action

An action that identifies the game has reached the end.
Start and End actions should be used ONLY in the mail loadable action file!

```
{
    "parseIdentifier": "END",
    "attributes": {}
}
```

Available Attributes

No attributes are needed.

Restart Action

An action that restarts the entire action hierarchy!

```
{
    "parseIdentifier": "RESTART",
    "attributes": {}
}
```

Available Attributes

No attributes are needed.

Play Audio

An action that plays an audio.

```
{
    "parseIdentifier": "PLAY_AUDIO",
    "attributes": {
        "path": "\root\audio_folder\audio.ogg"
    }
}
```

Parameter	Type	Description
path	`required`	path should be a path for a useful audio: `\root\audio_folder\audio.ogg`

Show Color

An action that shows a color on a dot.
Colors should be specified if different colors should be applied to different dots.
Color should be specified wether it is needed to update all colors at once.
One of those properties MUST be specified!

```
{
    "parseIdentifier": "SHOW_COLOR",
    "attributes": {
        "fade": true,
        "colors": [
            {
                "index" : 0,
                "color" : {
                    "red":255,
                    "green":255,
                    "blue":255
                }
            },
            {
                "index" : 1,
                "color" : {
                    "red":128,
                    "green":128,
                    "blue":128
                }
            }
        ],
        "color": {
            "red":255,
            "green":255,
            "blue":255
        }
```

```
    }
}
```

Parameter	Type	Description
colors	`required` *one*	`color` defines the next color to show, `index` defines the dot that will show the color
color	`required` *one*	color sets all the dots to the same color
fade	`optional`	fade animates the transation between the previous color and the taget ones

Play Animation Action

An action that plays an animation on the dots!
If no timeout is specified in the action's definition the action starts and ends immediately so make sure to set the `stops_at_end` key accordingly to your needs!

```
{
    "parseIdentifier": "PLAY_ANIMATION",
    "attributes": {
        "animation": "RAINBOW_CYCLE",
        "animation_affect_dots": true,
        "stops_at_end": true
    }
}
```

Available Attributes

Parameter	Type	Description
animation	`required`	animation should be one of: `RAINBOW`, `RAINBOW_CYCLE`, `THEATER_CHASE_RAINBOW`
animation_affect_dots	`optional`	wether animation animates the single leds or the entire dot
stops_at_end	`required`	when action finishes set `true` if the animation should stop, otherwise set `false`

Dynamic Load Action

An action that loads other actions from file and appends them after itself. Then goes directly to the next one.

```
{
    "parseIdentifier": "DYNAMIC_LOAD",
    "attributes": {
        "paths": [
            "config/blabla.action.json"
        ]
    }
}
```

Available Attributes

Parameter	Type	Description
paths	`required`	array of strings identifying one file each

Jump Action

An action that jumps to another action given its identifier. Notice that the identifier of the other action MUST exist and be present in the current loaded actions hierarchy. Just specify it as you can see here!

```
{
    "parseIdentifier": "JUMP",
    "attributes": {
        "actionIdentifier": "__ID__"
    }
}
```

Available Attributes

Parameter	Type	Description
actionIdentifier	`required`	string that identifies another action

Wait Input Action

An action that waits for a specified input (all of the specified are tapped) to proceed to the next action.
if `available_dot_indexes` is not specified than it waits for one input to proceed.

```
{
    "parseIdentifier": "WAIT_INPUT",
    "attributes": {
        "available_dot_indexes": [
            0,
            1
        ]
    }
}
```

Available Attributes

Parameter	Type	Description
available_dot_indexes	`optional`	array of integers identifying dots that are allowed to be tapped

Delay

A delay between 2 action.
Timeout should be in seconds.

```
{
    "parseIdentifier": "DELAY",
    "attributes": {
        "delay": 10
    }
}
```

Parameter	Type	Description
delay	`required`	delay in seconds between 2 actions

Combined Action

Defines multiple action to be executed together.

```
{
    "parseIdentifier": "COMBINED",
    "timeout": 10,
    "attributes": {
        "actions": [
            {
                "parseIdentifier": "PLAY_ANIMATION",
                "attributes": {
                    "animation": "RAINBOW_CYCLE",
                    "animation_affect_dots": true,
                    "stops_at_end": true
                }
            },
            {
                "parseIdentifier": "PLAY_AUDIO",
                "attributes": {
                    "path": "\root\audio_folder\audio.ogg"
                }
            }
        ],
        "policy" : "EXIT_TIMEOUT"
    }
}
```

Available Attributes

Parameter	Type	Description
actions	`required`	actions to be executed together. Actions such as `DYNAMIC_LOAD`, `END`, `START`, `RESTART`, `JUMP` are automatically ignored if added!
policy	`required`	two types of policies: `WAIT_ALL` (waits all actions to be terminated before continuing), `WAIT_FIRST` (when first is completed go to the next action), `EXIT_TIMEOUT` (kills this entire group when timeout occurs)

Single Selection Menu Action

A menu which makes people choose between different options. When one is selected its actions are executed!

Options MUST contain as many options as dots are in the system!

```
{
    "parseIdentifier": "SINGLE_CHOICE_MENU",
    "timeout": 10,
    "attributes": {
        "options": [
            {
                "actions" : [],
                "color" : {
                    "red" : "25",
                    "green" : "25",
                    "blue" : "25"
                }
            }
            {
                "actions" : [],
                "color" : {
                    "red" : "25",
                    "green" : "25",
                    "blue" : "25"
                }
            }
            {
                "actions" : [],
                "color" : {
                    "red" : "25",
                    "green" : "25",
                    "blue" : "25"
                }
            }
            {
                "actions" : [],
                "color" : {
                    "red" : "25",
                    "green" : "25",
                    "blue" : "25"
                }
            }
            {
                "actions" : [],
                "color" : {
                    "red" : "25",
                    "green" : "25",
                    "blue" : "25"
                }
            }
        ],
        "timeout_actions": []

    }
}
```

Parameter	Type	Description

options	`required`	options available: `actions` is an array of action objects, `color` is the color associated with that action! Options MUST contain as many options as dots are in the system!
timeout_actions	`required`	action object array to be executed after a timeout

---

Multiple Choice Menu Action

Defines a menu with different options and when one sequence is correctly inserted than it triggers its actions, otherwise it triggers a wrong choices defined action!

Options MUST contain as many options as dots are in the system!

```
{
    "parseIdentifier": "MULTIPLE_CHOICE_MENU",
    "timeout": 10,
    "attributes": {
        "options": [
            { "red": 255, "green": 255, "blue": 255 },
            { "red": 255, "green": 255, "blue": 255 },
            { "red": 255, "green": 255, "blue": 255 },
            { "red": 255, "green": 255, "blue": 255 }
        ],
        "allowed_sequences": [
            {
                "actions": [],
                "chosen_options": [0, 3]
            }
        ],
        "wrong_sequence_actions": [],
        "abort_actions": [],
        "time_between_choices": 1
    }
}
```

Available Attributes

Parameter	Type	Description
options	`required`	contains an array of colors that will be shown on the dots as choices. The number of options MUST be equal to the number of dots!

allowed_sequences	`required`	array of items containing a list of actions to be executed when whose options are selected!, `chosen_options` should be an array of integers which reflects the `options` described above. Remember: order matters!
wrong_sequence_actions	`required`	Sequence of actions that are executed if a not allowed sequence if inserted!
abort_actions	`required`	Sequence of actions that are executed when timeout fires and no choice is taken!
time_between_choices	`optional`	Maximum time allowed between each tap on dots! if timeout fires up than it checks the inserted sequence and does its things according to the correctness of it.