An IDE for Visual Programming of Smart Automations in the Internet of Things

Programming Languages
and Tools Laboratory

Computer Science Department
University of Crete

Outline

Smart Automations for Everybody

Blockly Studio

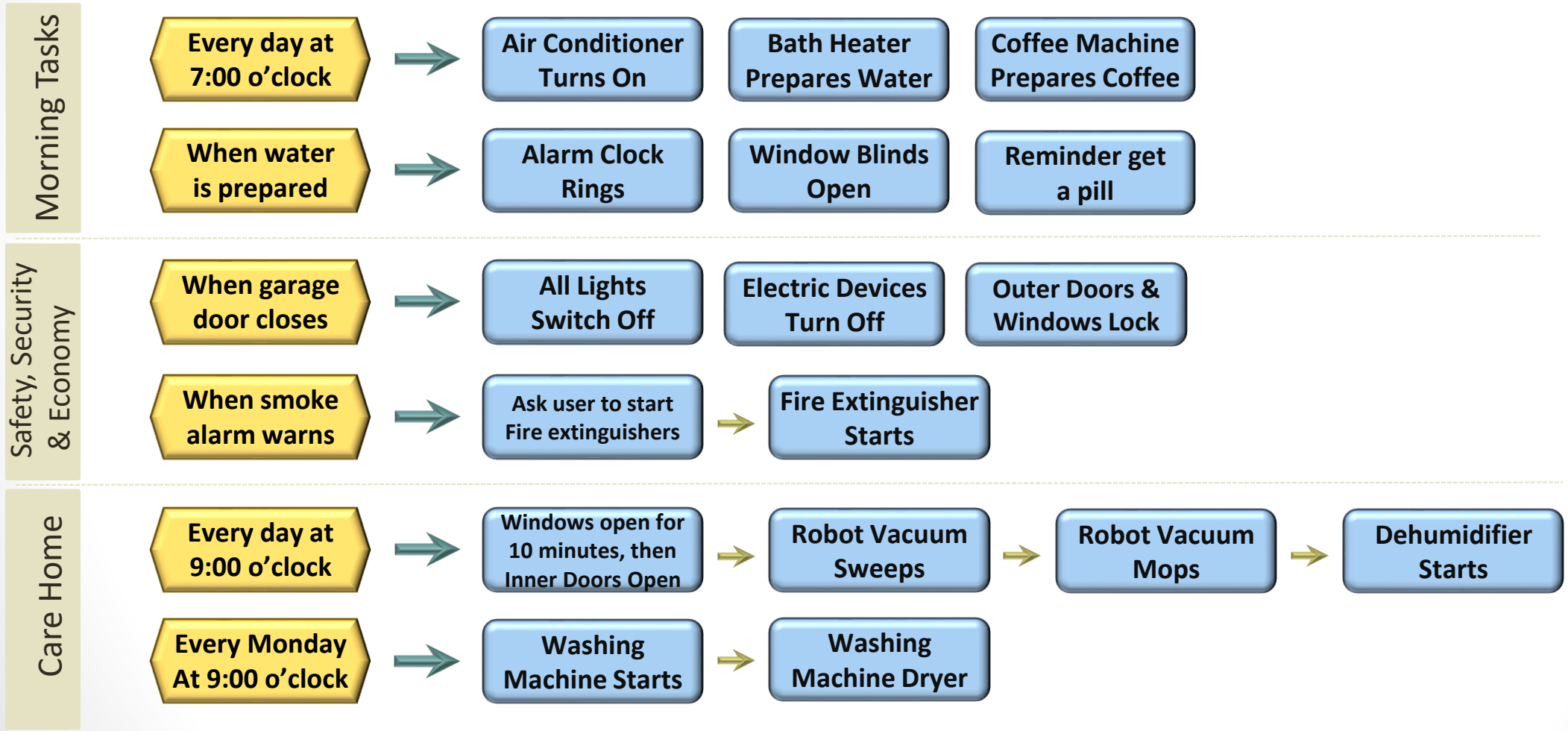
1. Projects and Editors
   - Device Management
   - Visual Programming for Smart Objects
   - Visual Programming for Automations
   - Project Management
   - Demo 1st – Building smart automations
2. Runtime and Debugger
   - Running Automations
   - User Interface for Smart Automations
   - Debugging Facilities for Smart Automations
   - Demo 2nd - Running and testing automations

Smart Automations for Everybody (1/3)

- Smart connected world in the IoT era
- People's daily lives could benefit through automations
  - Taking advantage of the smart objects
- Demands for such automations
  - Highly personalized and fluid
- Examples of applicable environments
  - Smart Environment/Living
    - Home, Office, School automations etc.
  - Ambient Assisted Living
  - Smart Tourism and Mobility
    - Transportation assistant

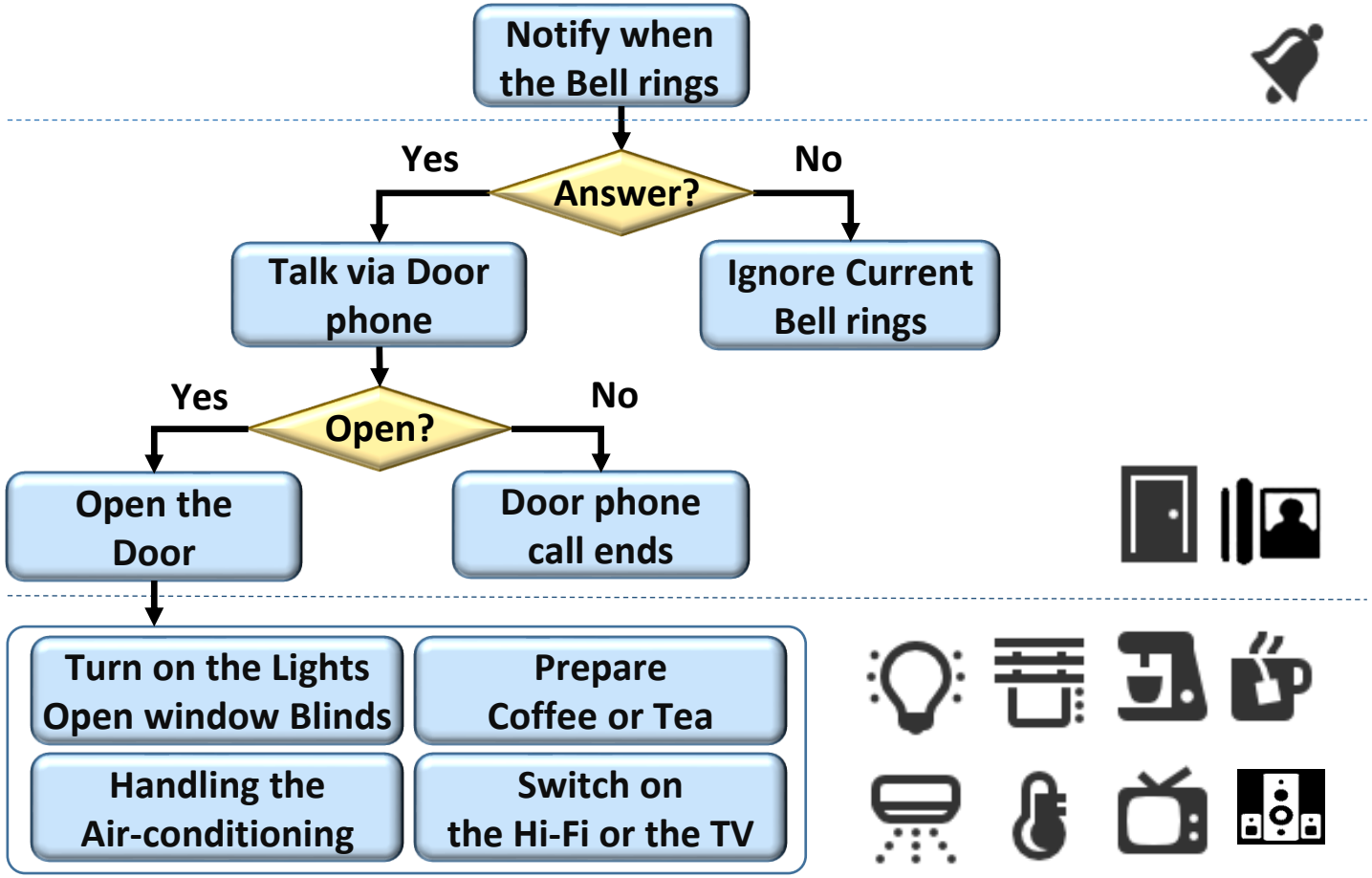
Smart Automations for Everybody (2/3)

Example: Daily automations at home



Smart Automations for Everybody (3/3)

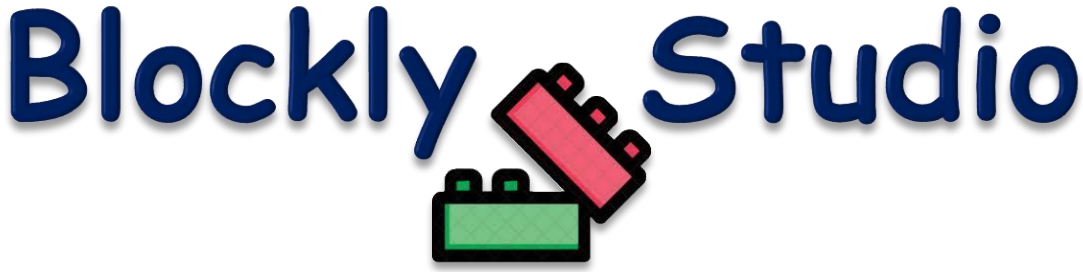
Scenario Example: Remote Hospitality



End-User Programming

Highly personalized nature of automations + *Typically small-scale micro-applications* → *Niche-market with unlike industrial interest*

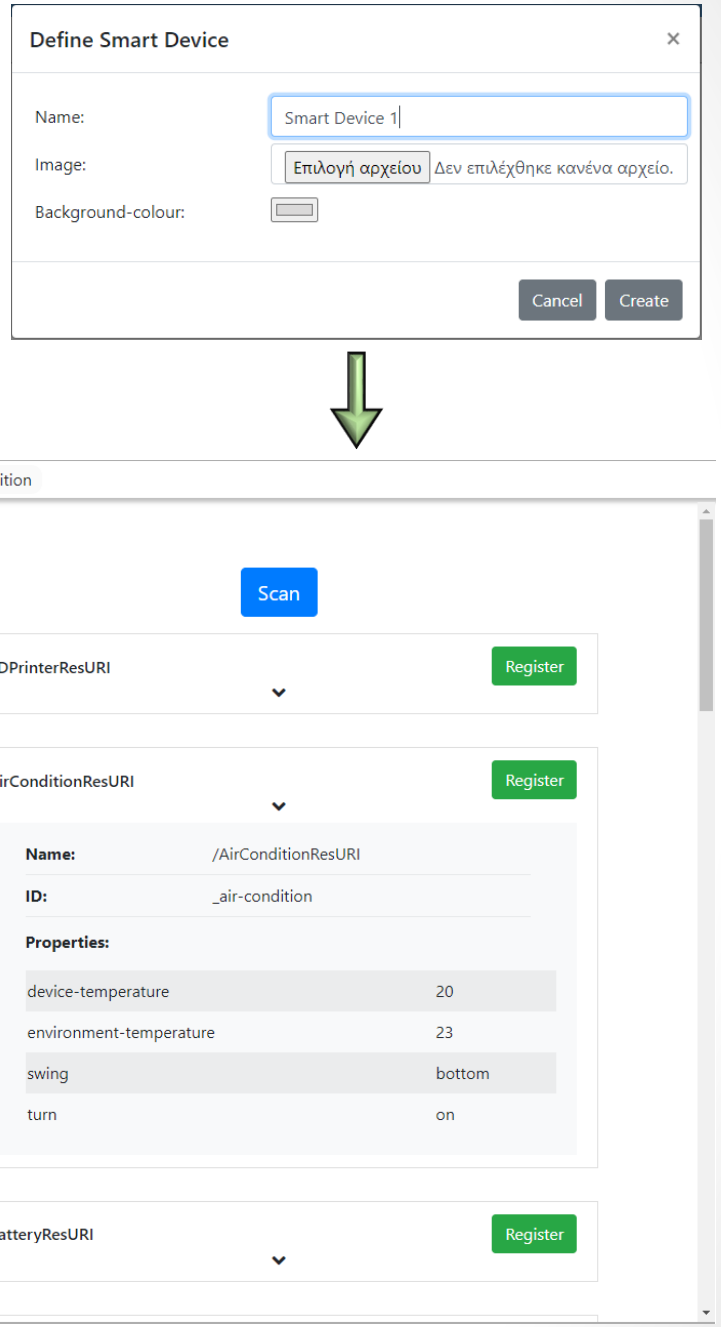
- End-users should be enabled to craft such automations
- End-user programming (EUP) is a vehicle to achieve this goal
  - Methods, techniques and tools that allow users (non professional developers) to create, modify, or extend a program
  - Existing techniques
    - Spreadsheets
    - Scripting Languages
    - Natural Languages
    - **Visual Programming**



1. Editors and Projects
2. Runtime and Debugger

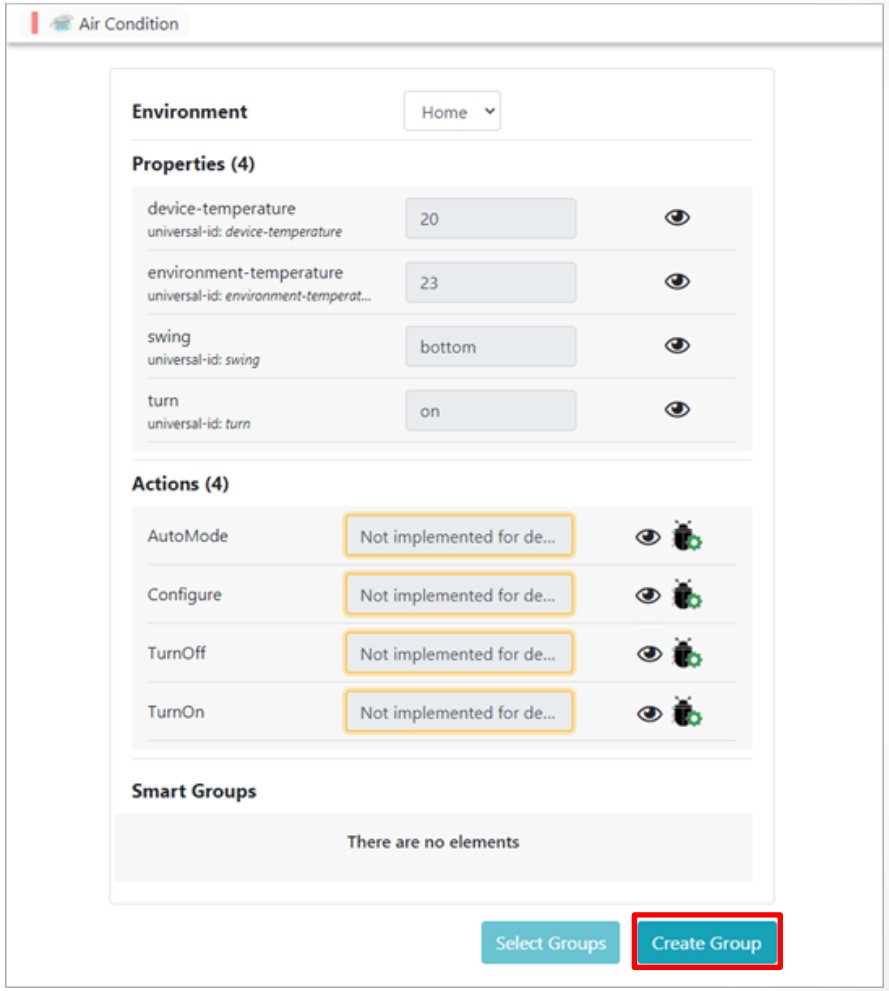
Device Management

- Handling *Smart Objects as micro services*
  - Customized VPL domain elements
- Communicating with smart objects via IoTivity



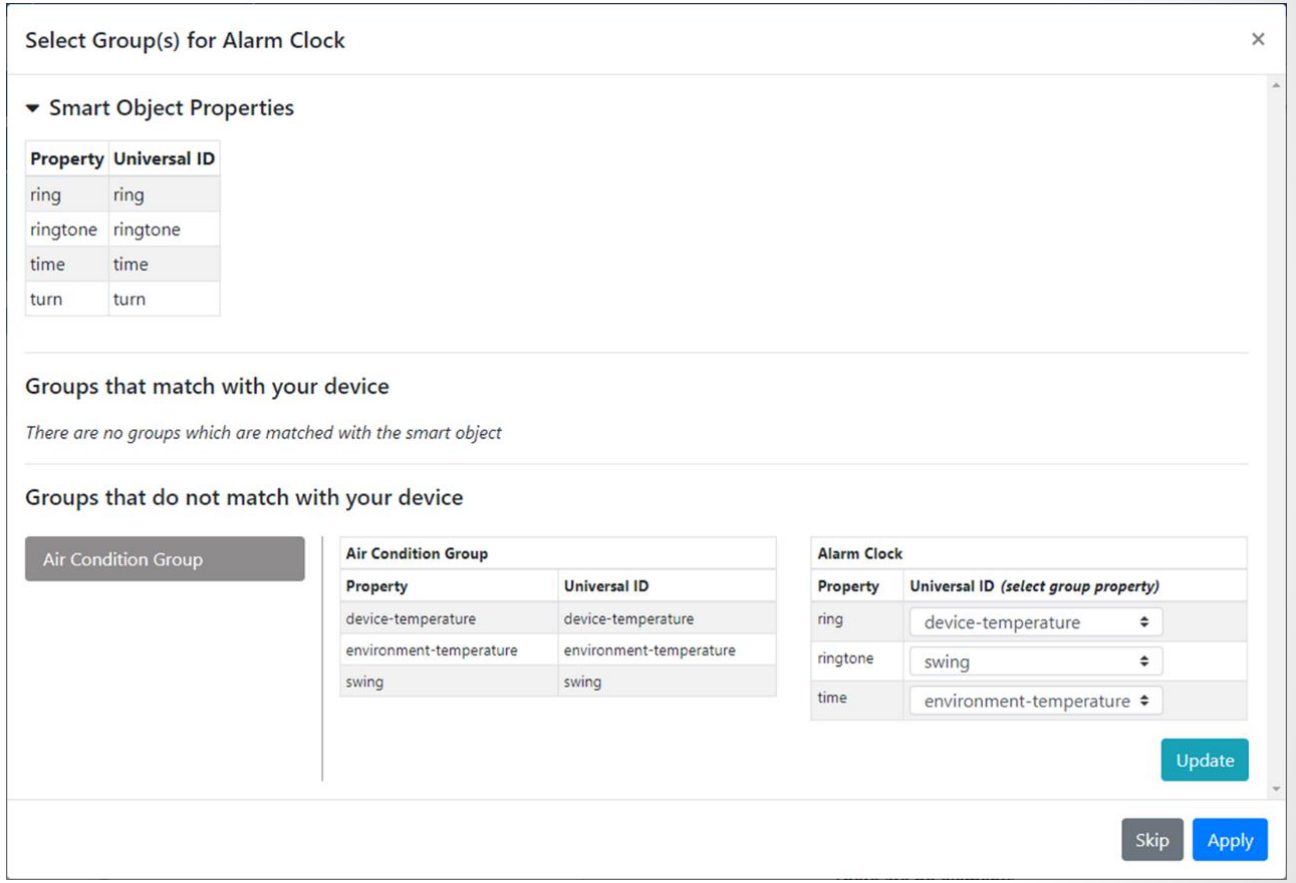
Device Management

- Handling *Smart Objects as micro services*
  - Customized VPL domain elements
- Communicating with smart objects via IoTivity
- Organizing smart objects in groups



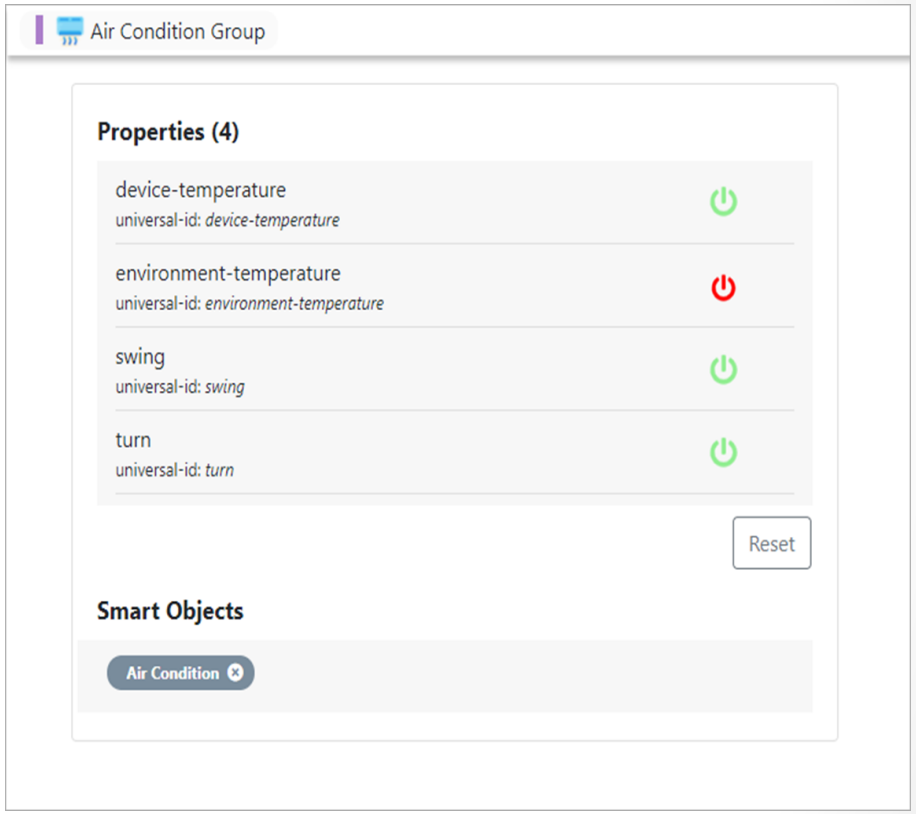
Device Management

- Handling *Smart Objects as micro services*
  - Customized VPL domain elements
- Communicating with smart objects via IoTivity
- Organizing smart objects in groups



Device Management

- Handling *Smart Objects as micro services*
  - Customized VPL domain elements
- Communicating with smart objects via IoTivity
- Organizing smart objects in groups

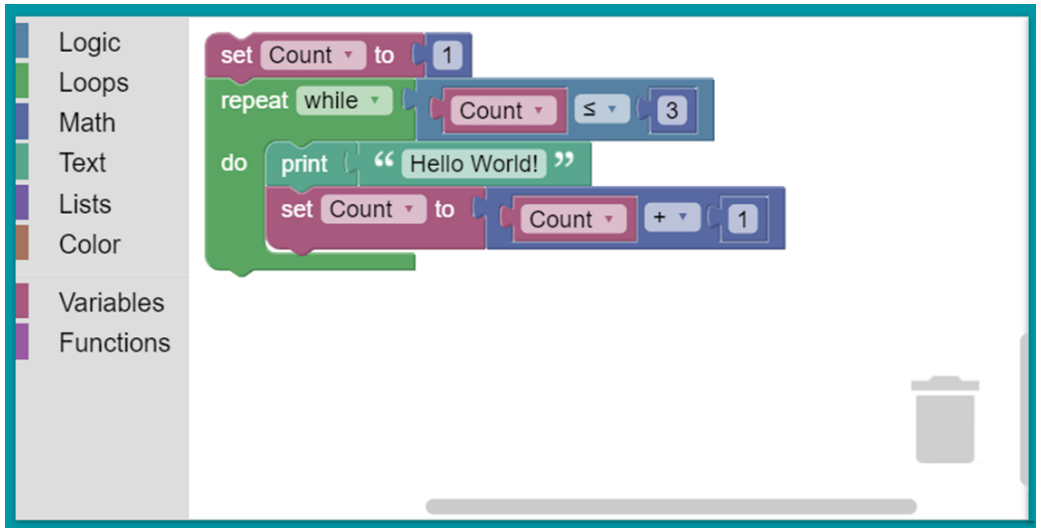


Device Management

- Handling *Smart Objects as micro services*
  - Customized VPL domain elements
- Communicating with smart objects via IoTivity
- Organizing smart objects in groups
- **Do not deal** with programming of their behavior
- Cooperating with IDE
  - Notifying about user actions in smart objects (create, edit, delete)

Visual Programming Editor

- General purpose editors
  - Basic programming expressions
    - Variables, Math & Logic Operations, Branches, Loops, etc.
  - We use *Blockly* library
    - Allows define new blocks per domain



- Defining new blocks to develop IoT applications
  - Blockly does **not deals** with dynamic handling (i.e. create, edit, delete) blocks during the EUD process
  - VPL Manager deals with it
    - Exchanging messages with IDE and creating, editing and deleting blocks

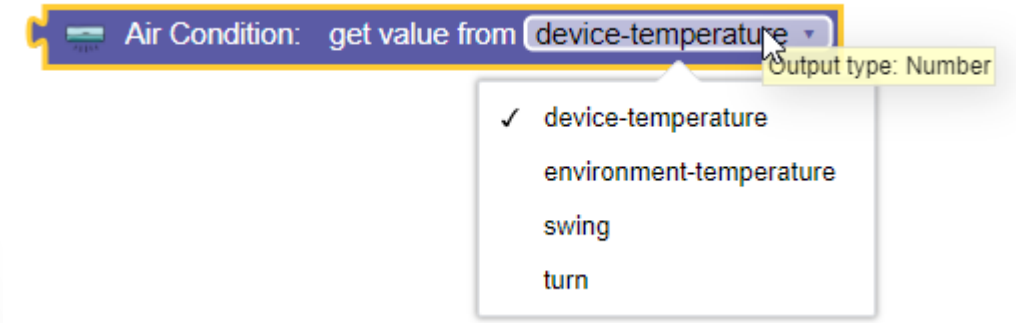
Visual Blocks for Devices

- Based on data are exported by editor for smart object
  - IDE’s extension mechanism provides updated the blocks of smart objects

Visual Blocks for Devices

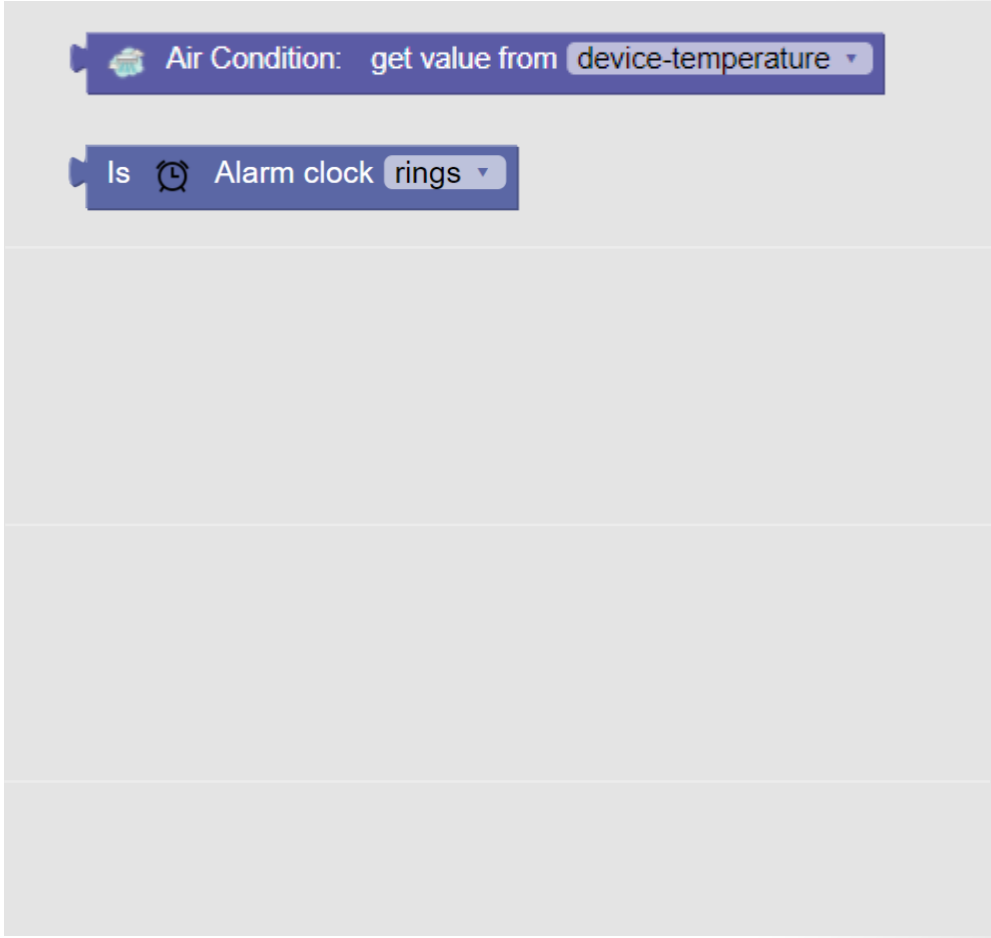
- Based on data are exported by editor for smart object
  - IDE's extension mechanism provides updated the blocks of smart objects

Getters



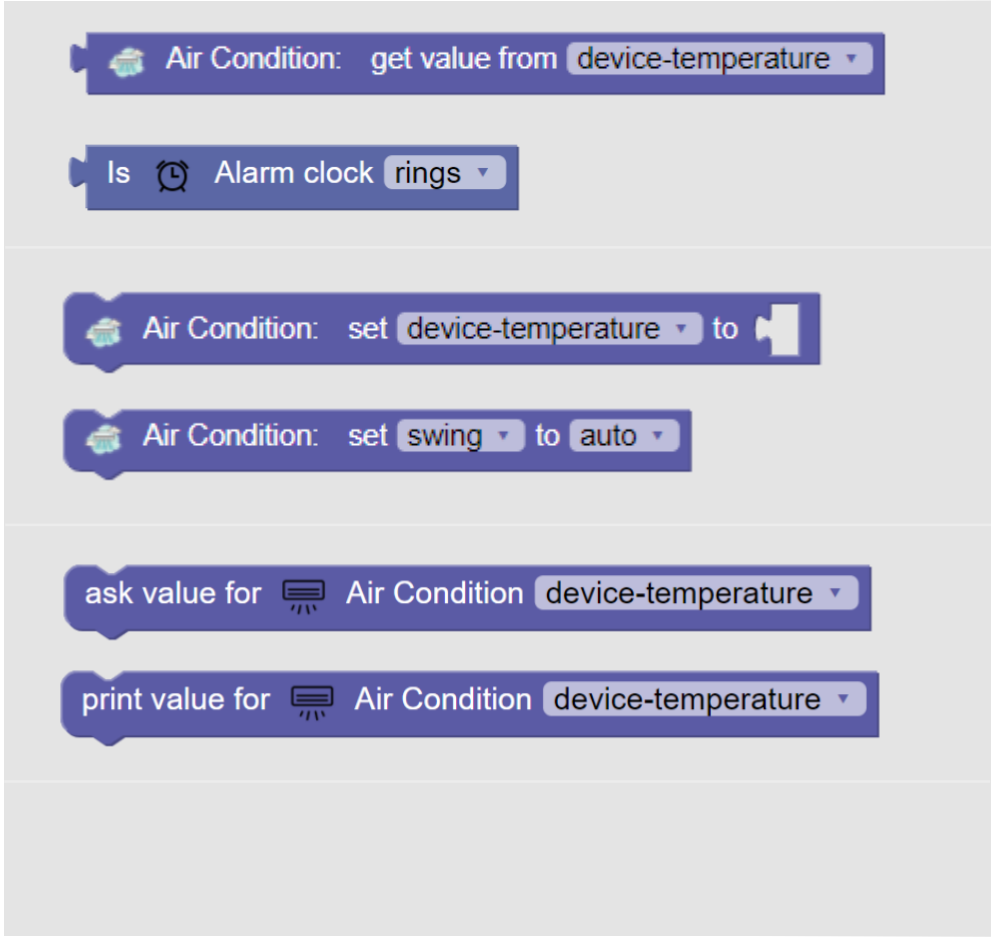
Visual Blocks for Devices

- Based on data are exported by editor for smart object
  - IDE's extension mechanism provides updated the blocks of smart objects



Visual Blocks for Devices

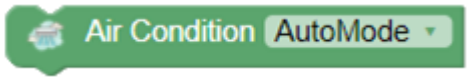
- Based on data are exported by editor for smart object
  - IDE's extension mechanism provides updated the blocks of smart objects

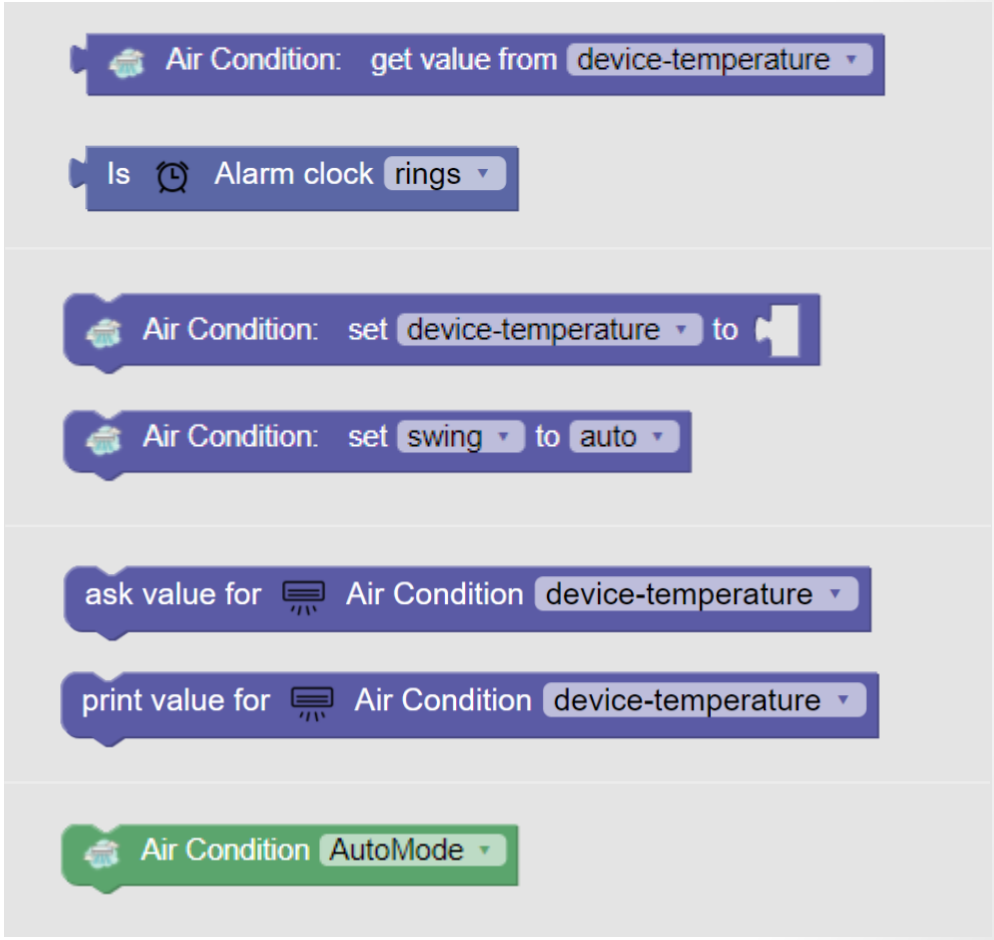


Visual Blocks for Devices

- Based on data are exported by editor for smart object
  - IDE's extension mechanism provides updated the blocks of smart objects

Actions

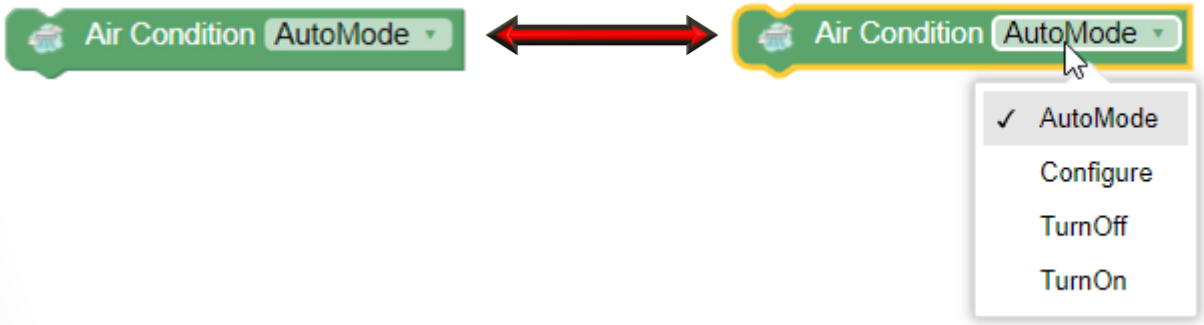


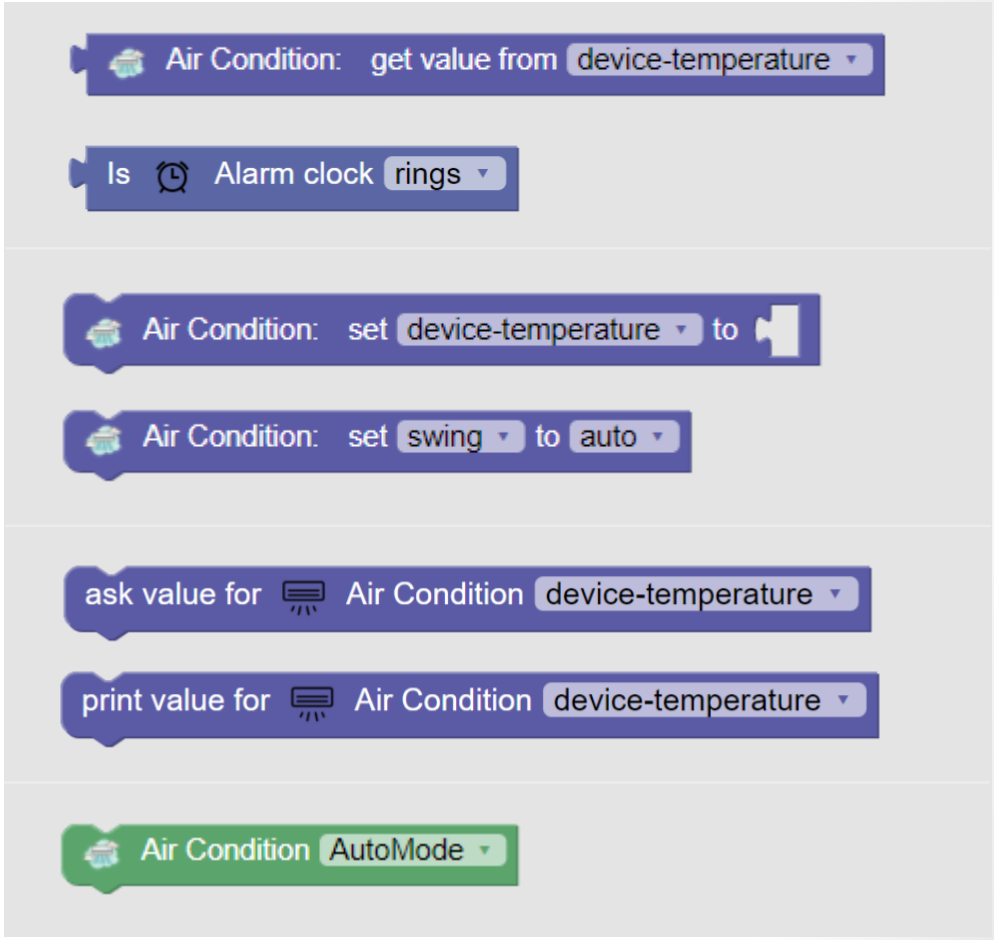


Visual Blocks for Devices

- Based on data are exported by editor for smart object
  - IDE's extension mechanism provides updated the blocks of smart objects

Actions

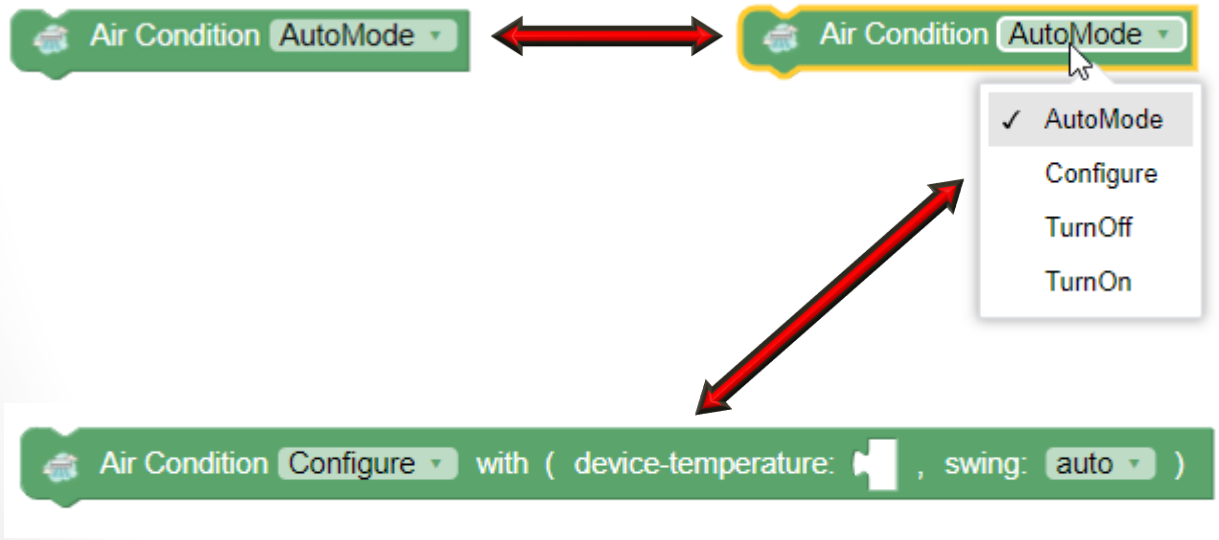


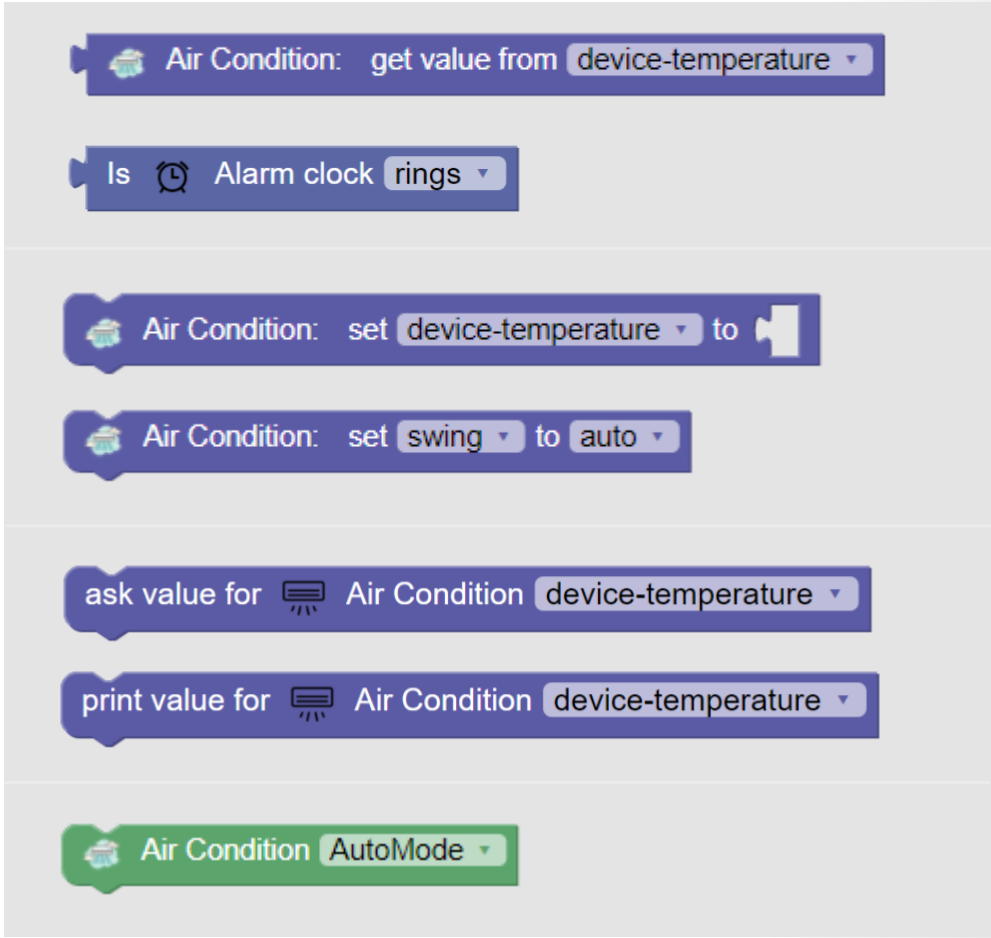


Visual Blocks for Devices

- Based on data are exported by editor for smart object
  - IDE's extension mechanism provides updated the blocks of smart objects

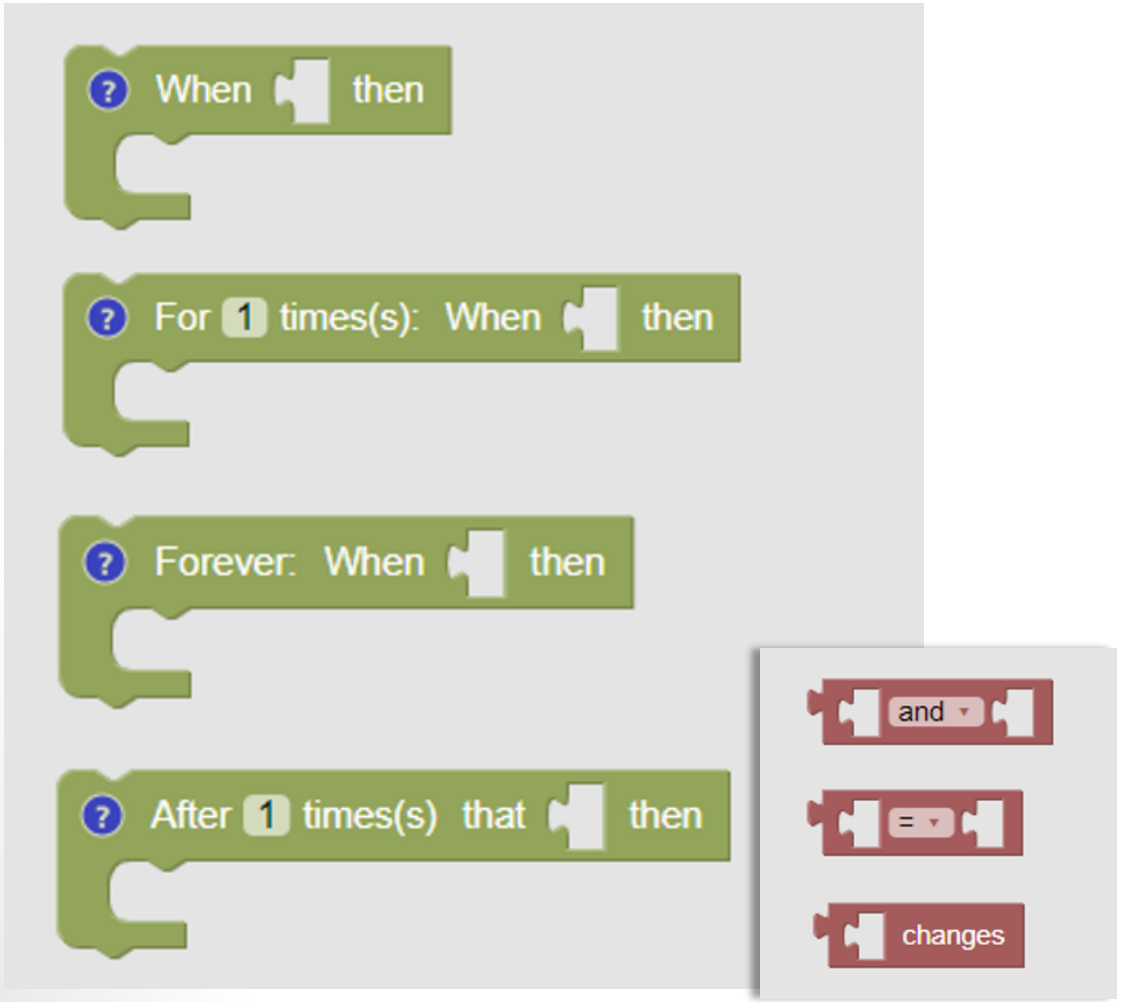
Actions



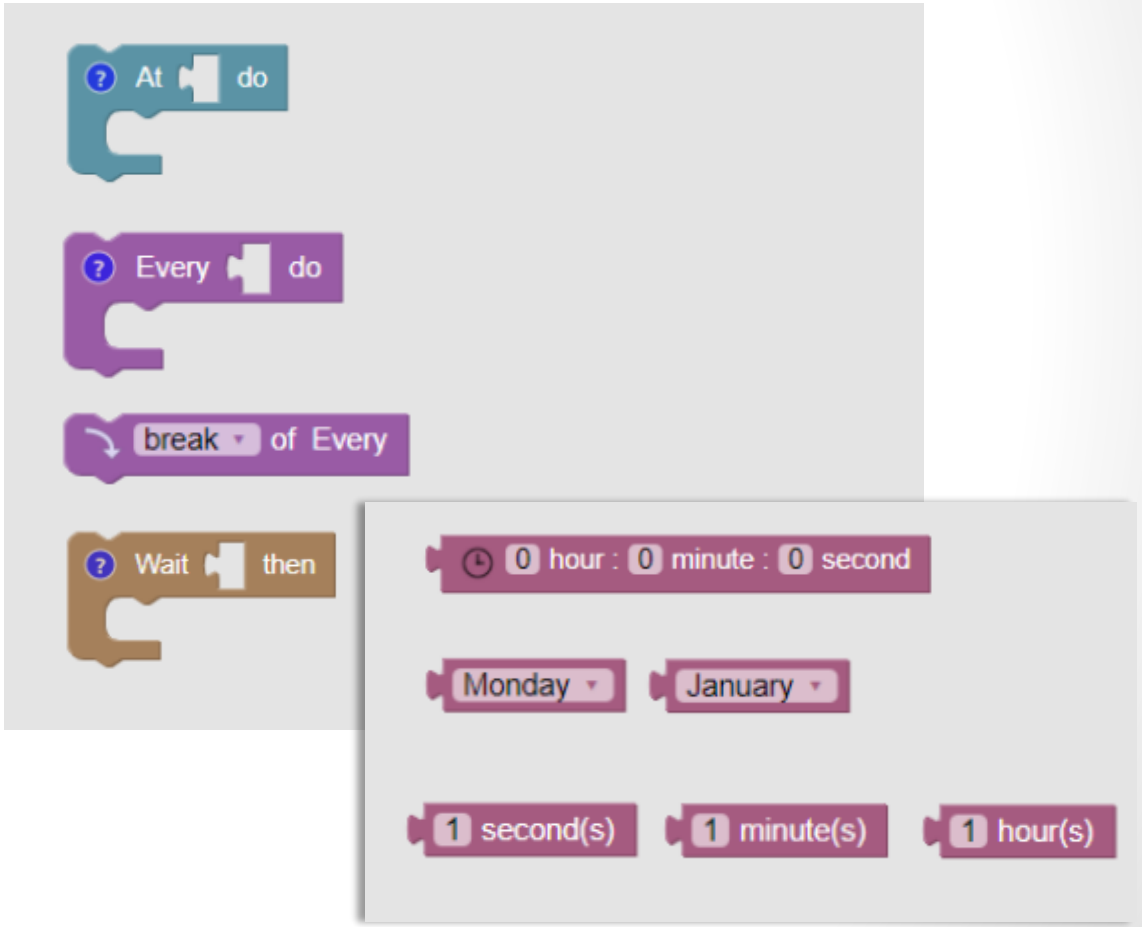


Visual Blocks for Smart Automations

Conditional Events

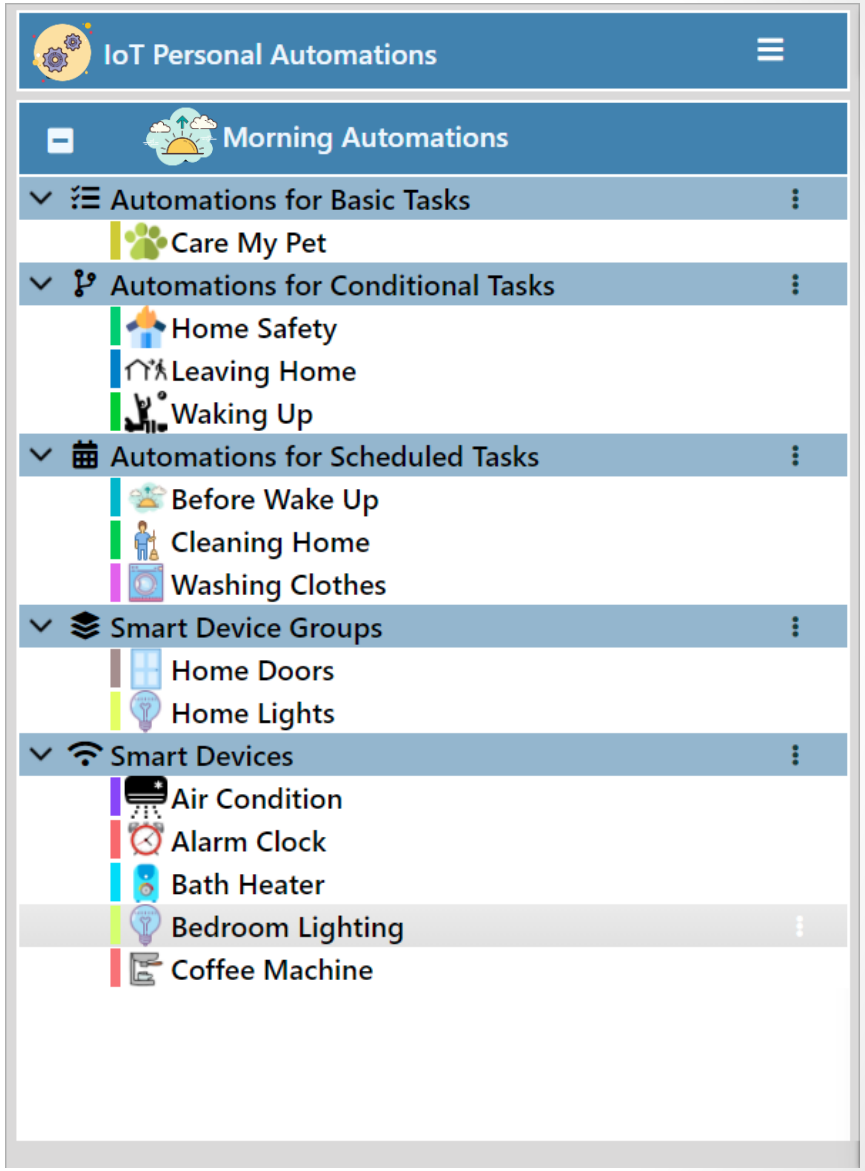


Scheduling Actions



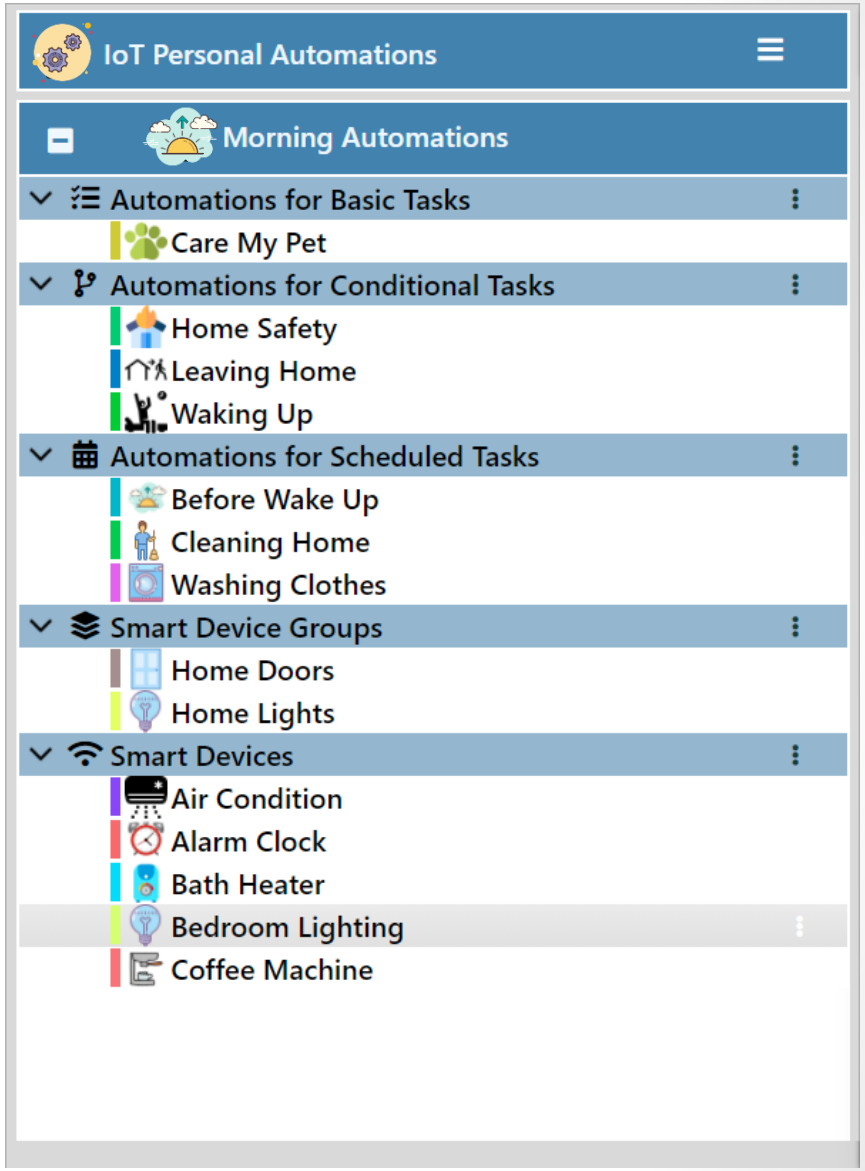
Project Management

- Other approaches focusing on learning experimentation than provide an IDE
  - As result, project manager is **not provided**



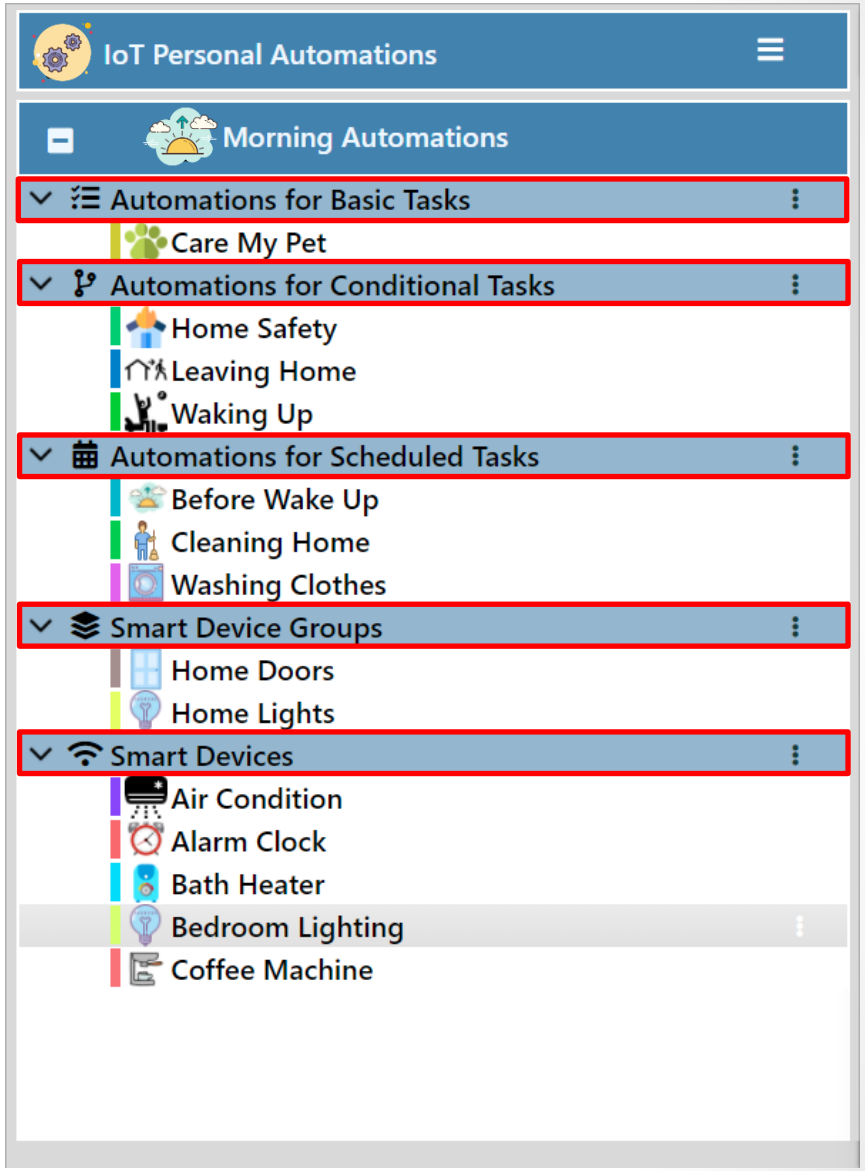
Project Management

- Other approaches focusing on learning experimentation than provide an IDE
  - As result, project manager is **not provided**
- Classic Project Manager of IDEs are not appropriate



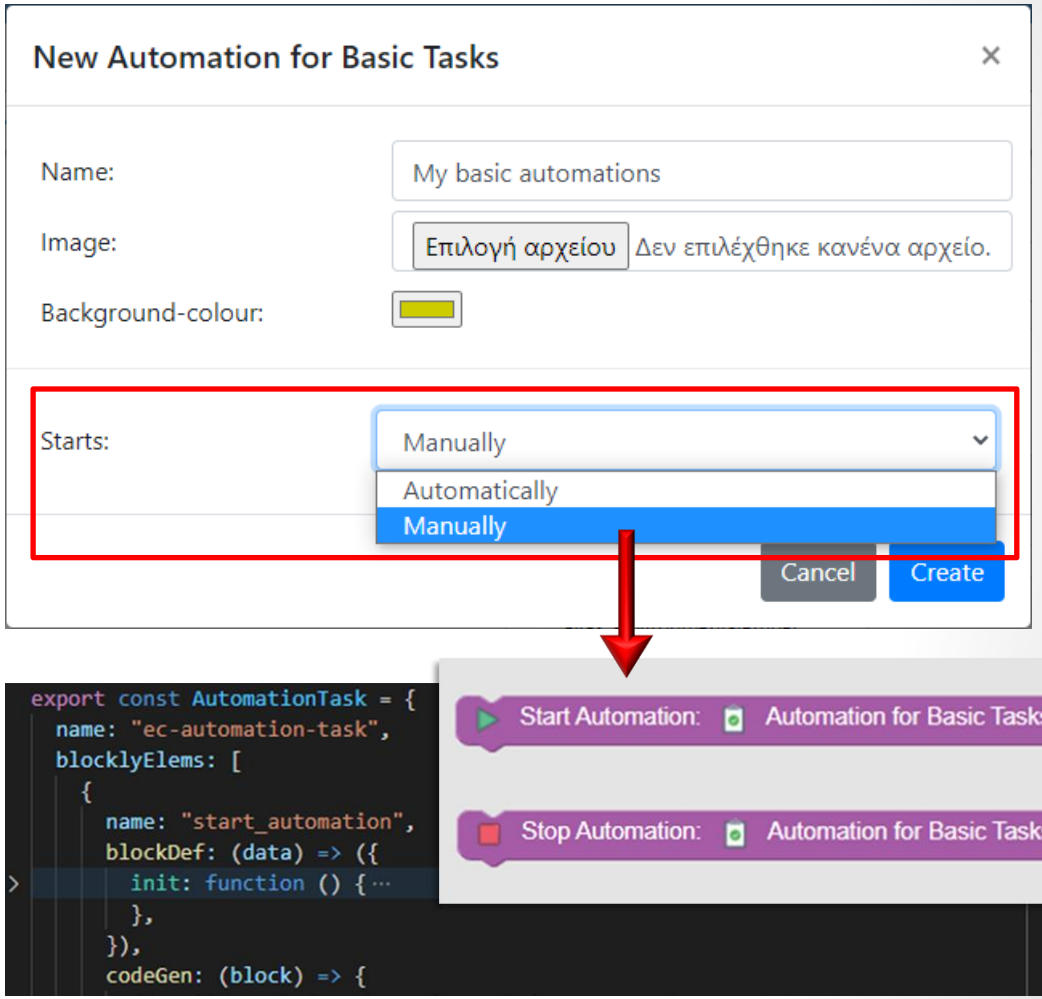
Project Management

- Other approaches focusing on learning experimentation than provide an IDE
  - As result, project manager is **not provided**
- Classic Project Manager of IDEs are not appropriate
  - Novices are not experienced on structuring projects
    - Assisted project structure is needed
    - Invent 5 types of project elements
    - Specified menu options and validation rules
    - Automatically handling dependencies of project elements

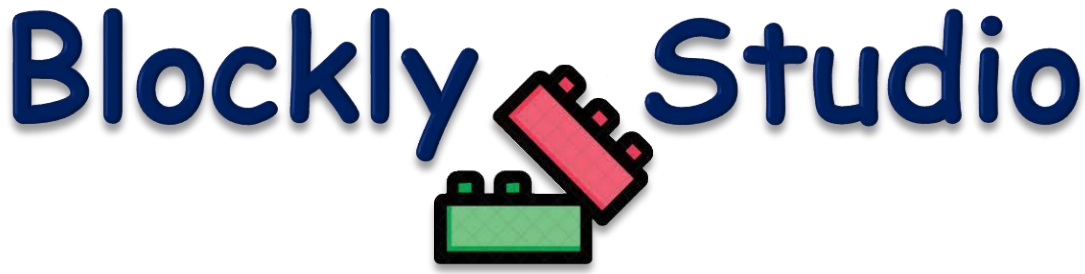


Project Management

- Other approaches focusing on learning experimentation than provide an IDE
  - As result, project manager is **not provided**
- Classic Project Manager of IDEs are not appropriate
  - Novices are not experienced on structuring projects
    - Assisted project structure is needed
    - Invent 5 types of project elements
    - Specified menu options and validation rules
    - Automatically handling dependencies of project elements
    - Enabling to choose when automations will begin through project element options



Demo Part 1/2:
Building Automations

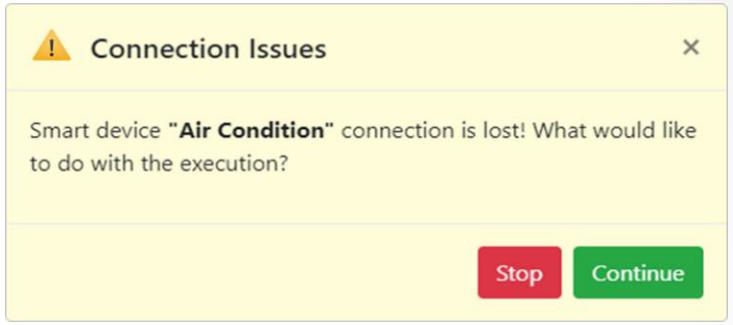


1. Editors and Projects

2. Runtime and Debugger

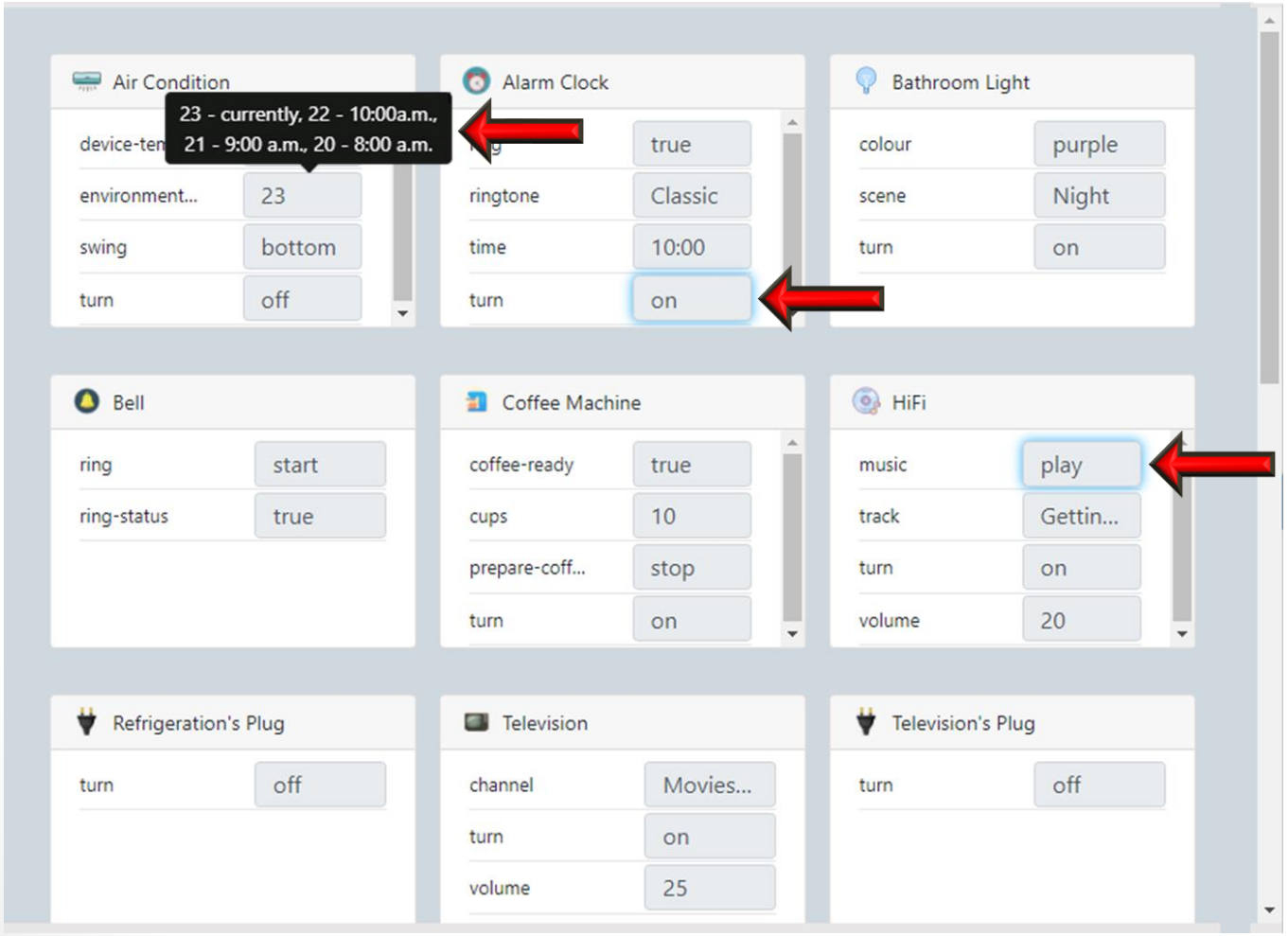
Running Smart Automations

- Handling failed requests for smart devices
  - Connection issues or not enable action
  - Notifying users via dialogues
- How end-users build UIs for their automations?
  - Using WYSIWYG editor
    - Difficult process
    - Extremely costly
    - Not practical for micro applications



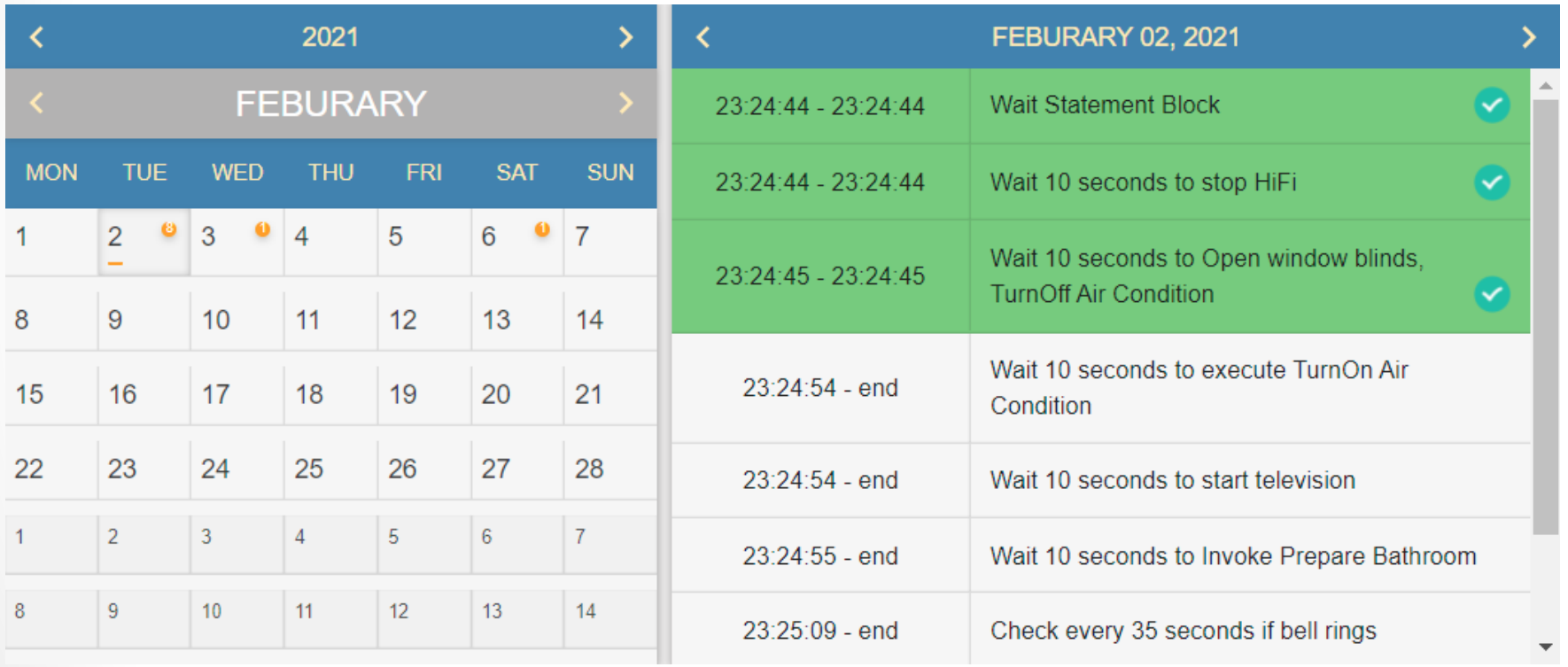
User Interface for Smart Automations

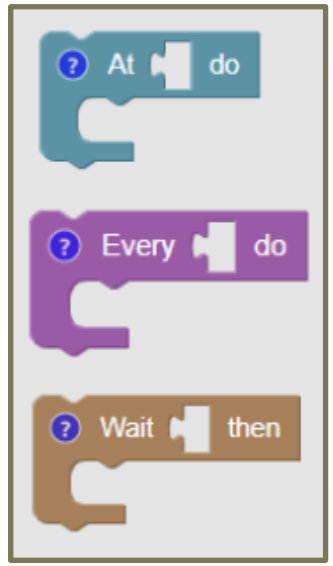
1. Smart Objects



User Interface for Smart Automations

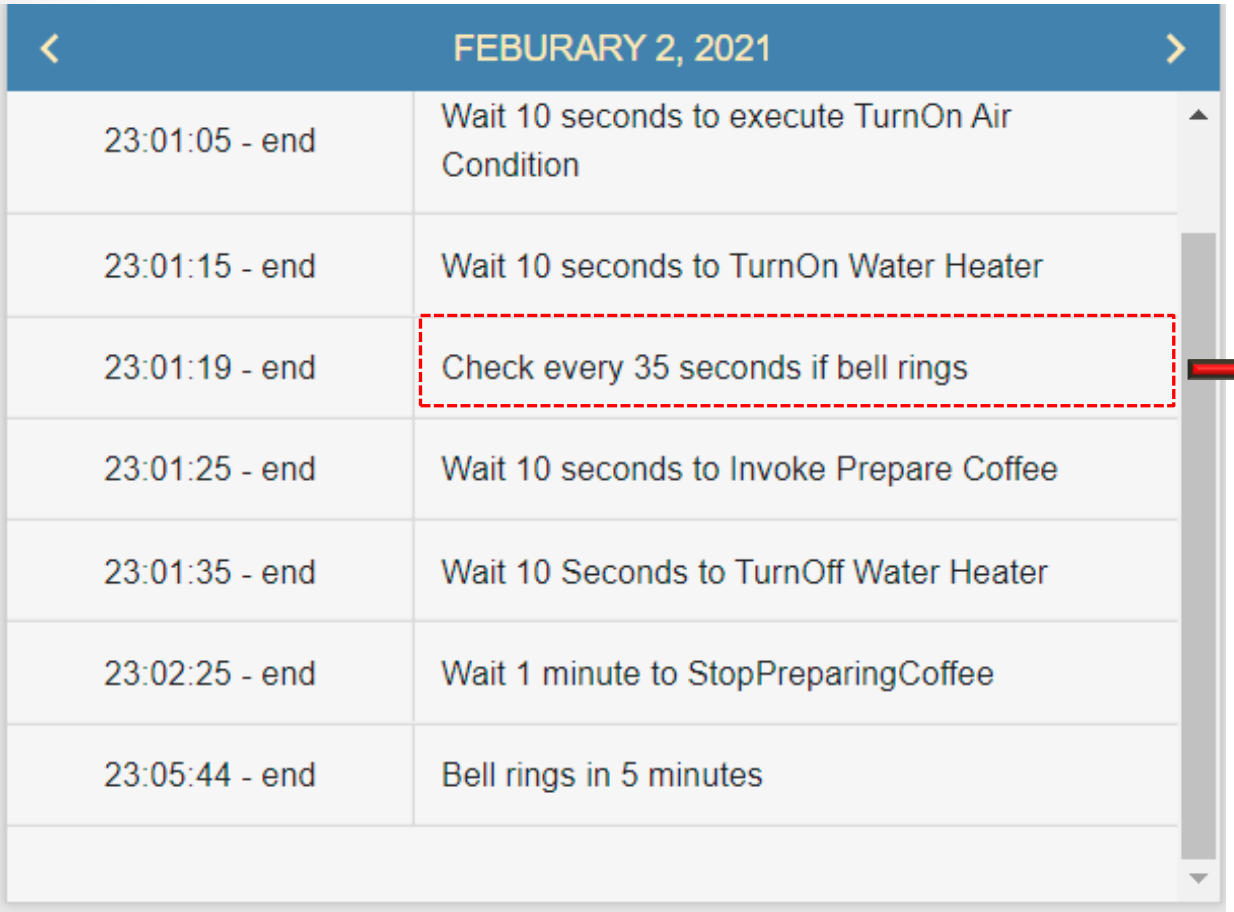
2. Scheduled Tasks

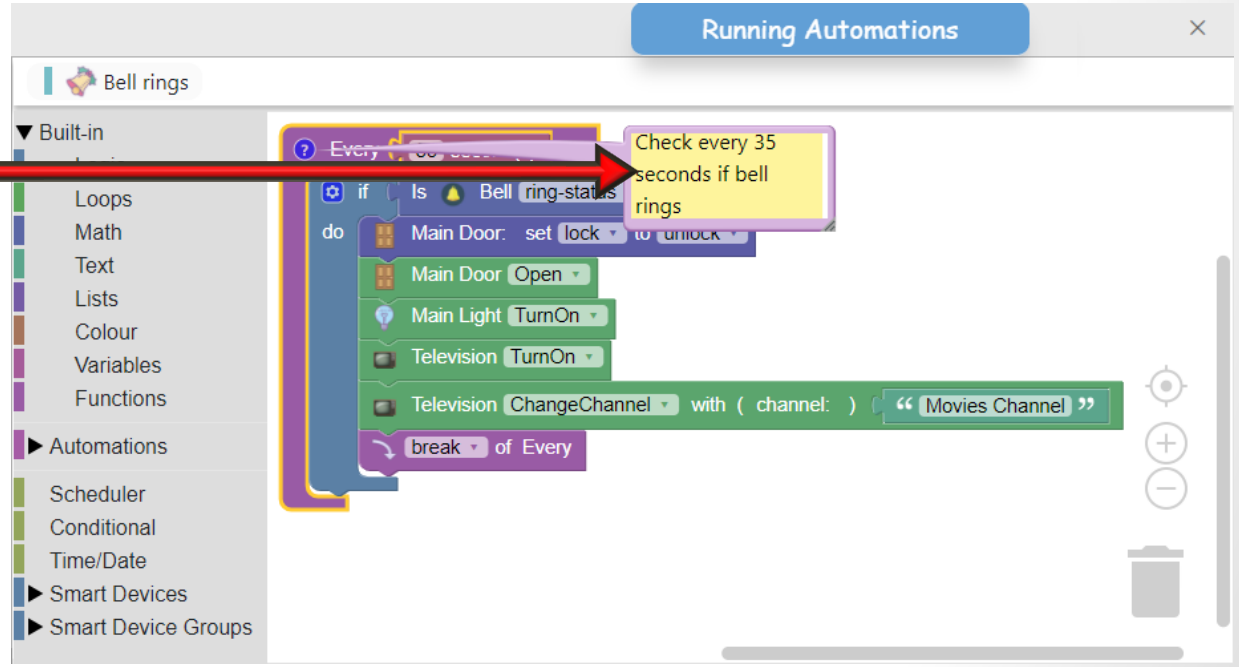




User Interface for Smart Automations

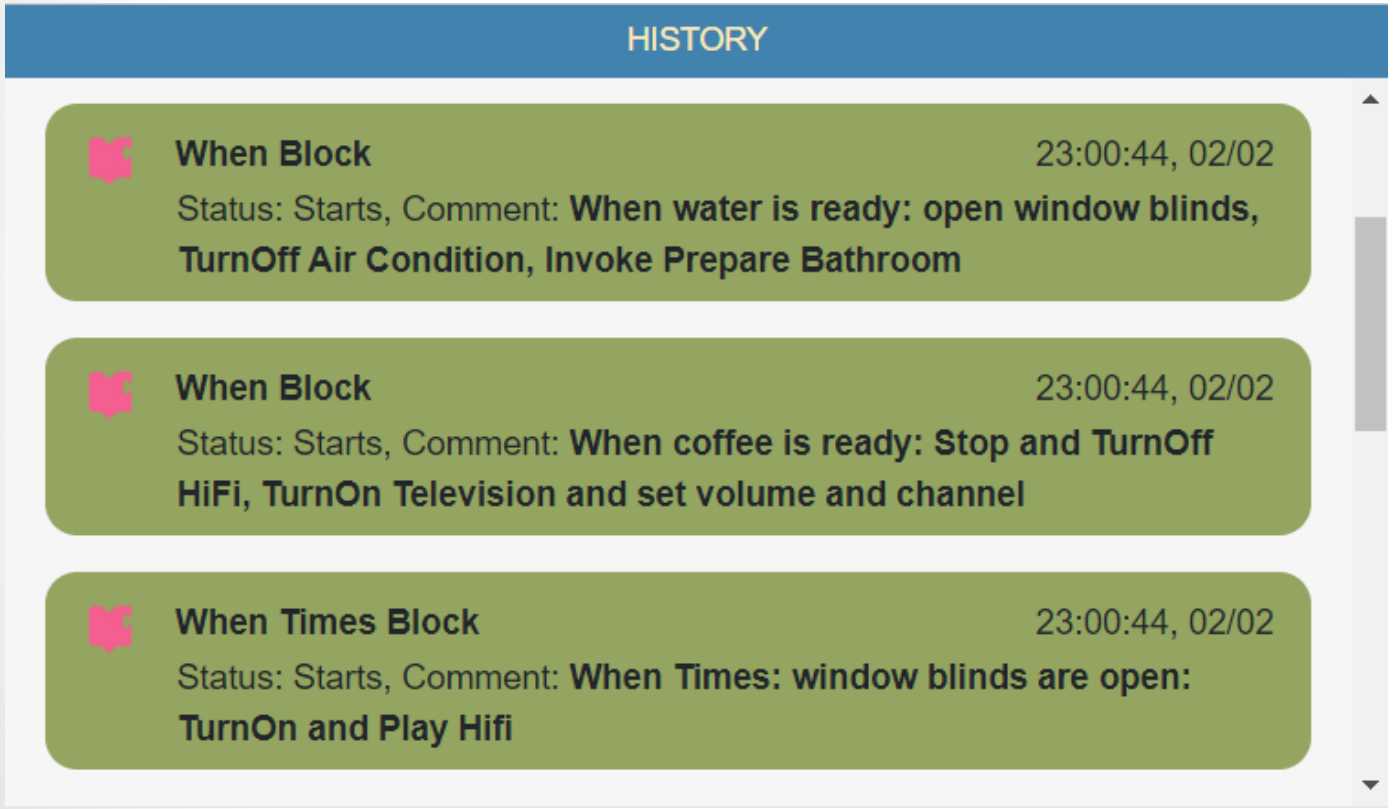
2. Scheduled Tasks





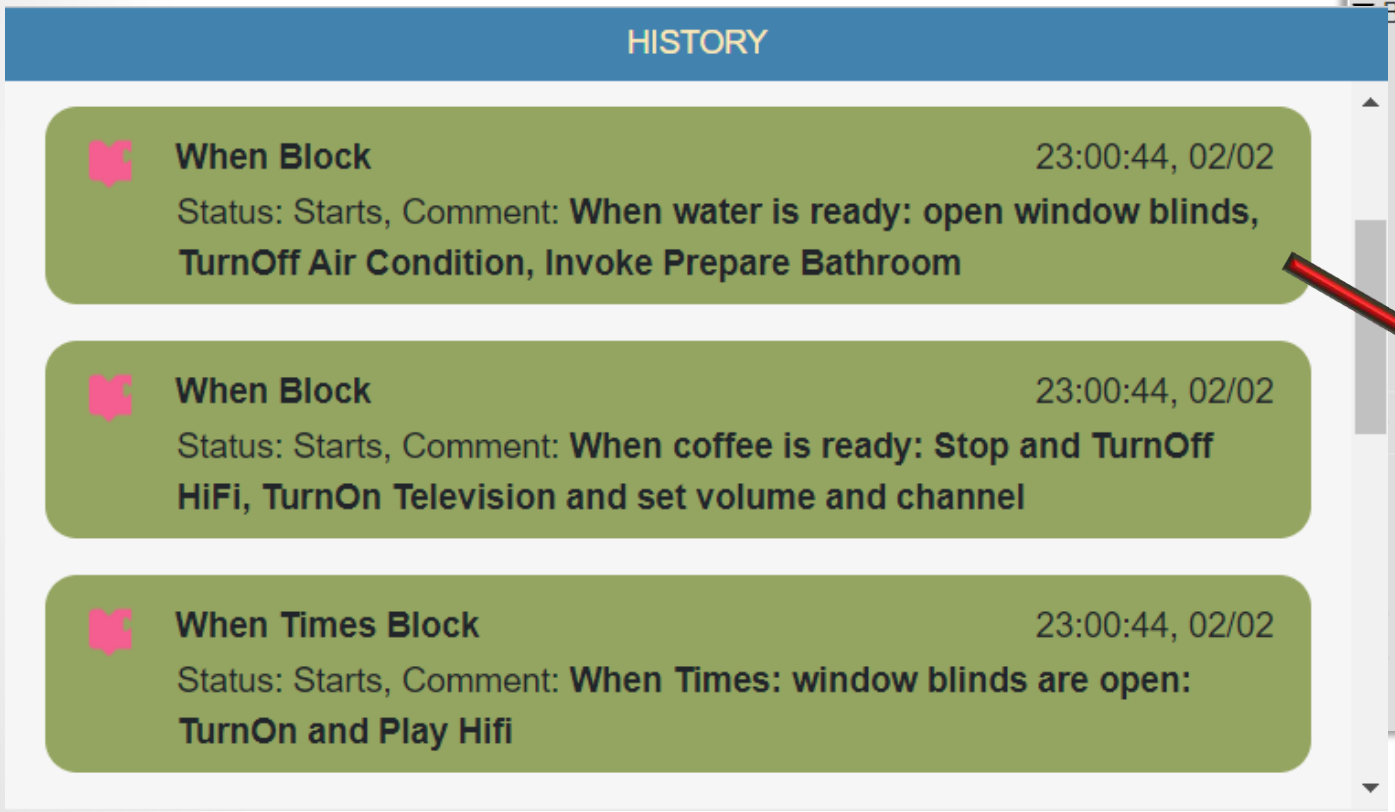
User Interface for Smart Automations

3. History View



User Interface for Smart Automations

3. History View

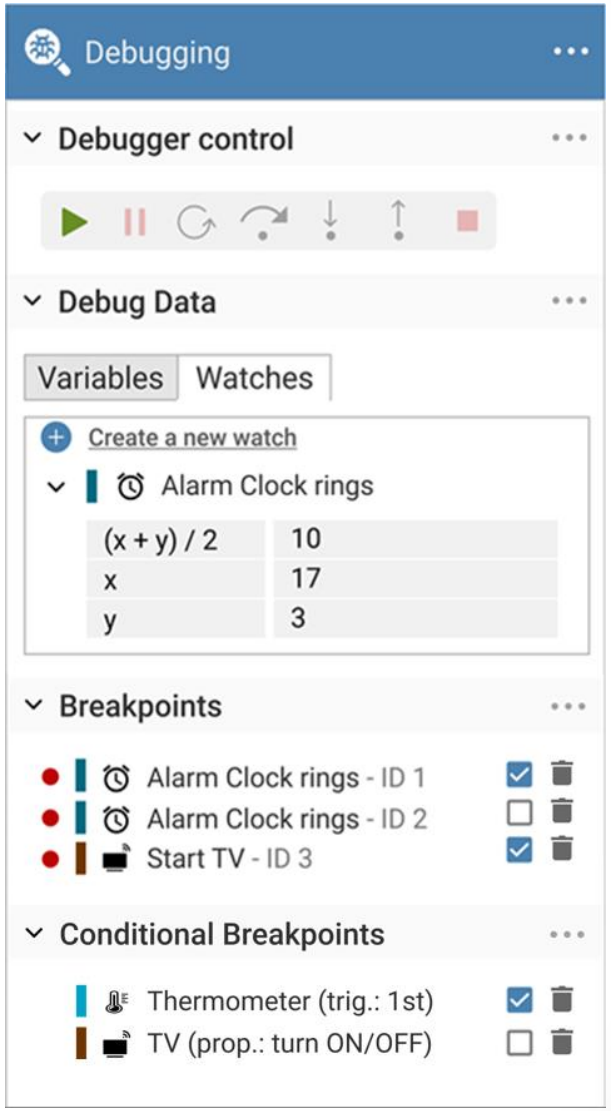


Why do Automations Occur?

- Users may wonder “what caused this automation to start?”
  - Not able to get information from source code executed
- Explanation notes of scheduled & conditional blocks
  - Inner blocks are able to change the execution code flow
  - **Not adequate to answer questions**
- Introducing new explanation blocks

Visual End-User Debugging

- Supporting full-scale visual debugger features
  - Tracing and watches
  - Breakpoints
  - Execution snapshots
  - Selecting project elements to execute

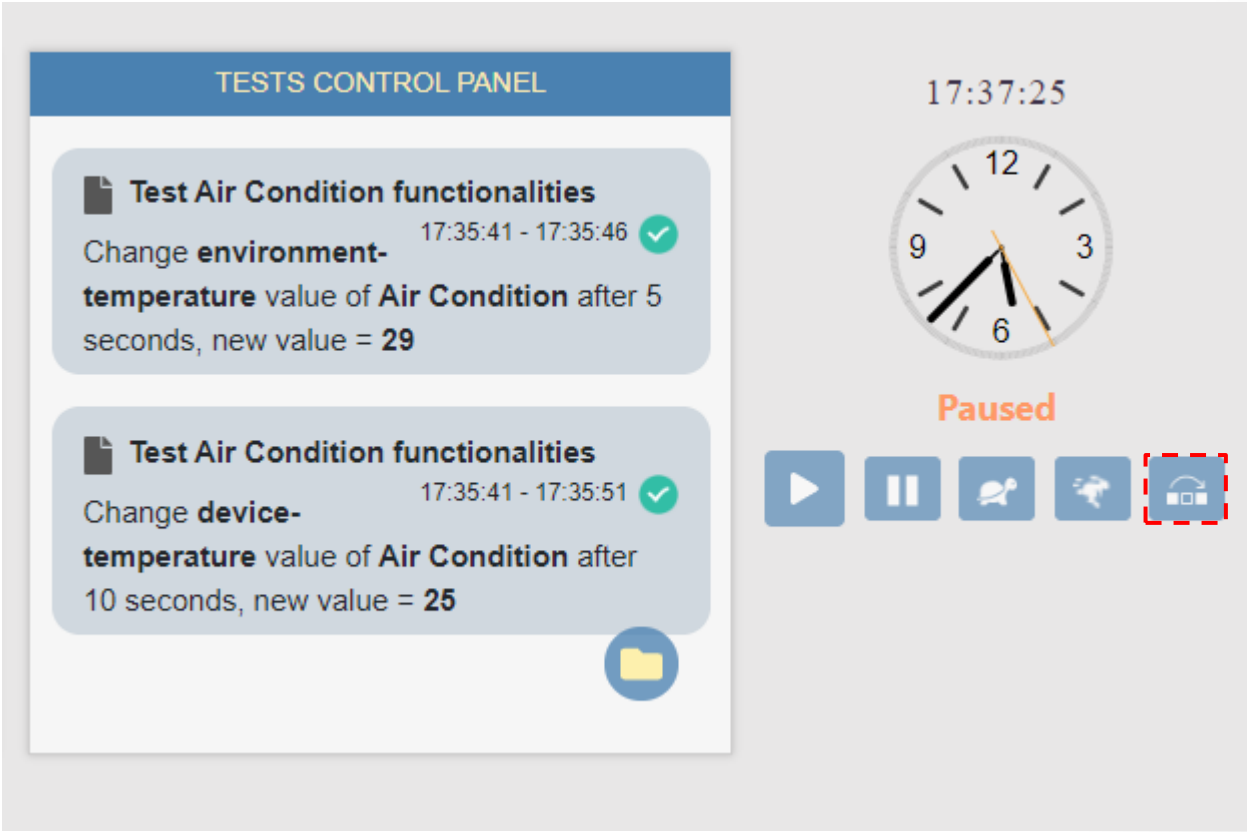


Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”

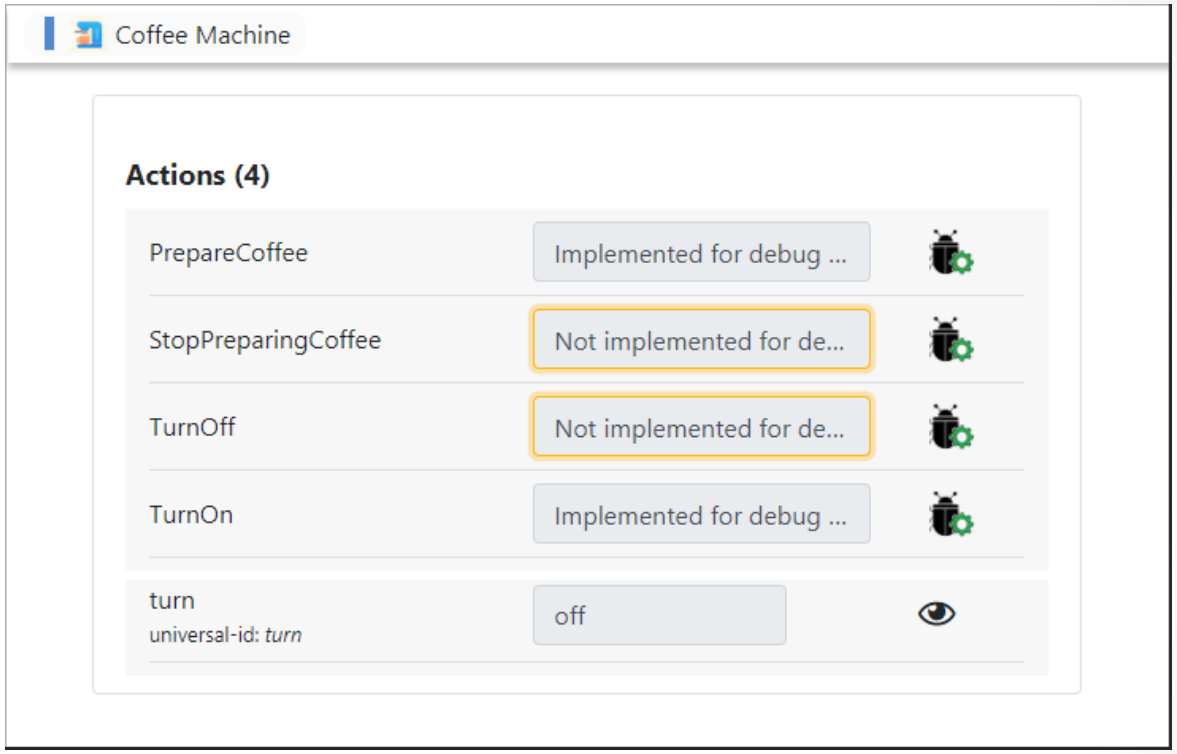
Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”
- Simulating smart environment
  - Smart devices
  - Date and time



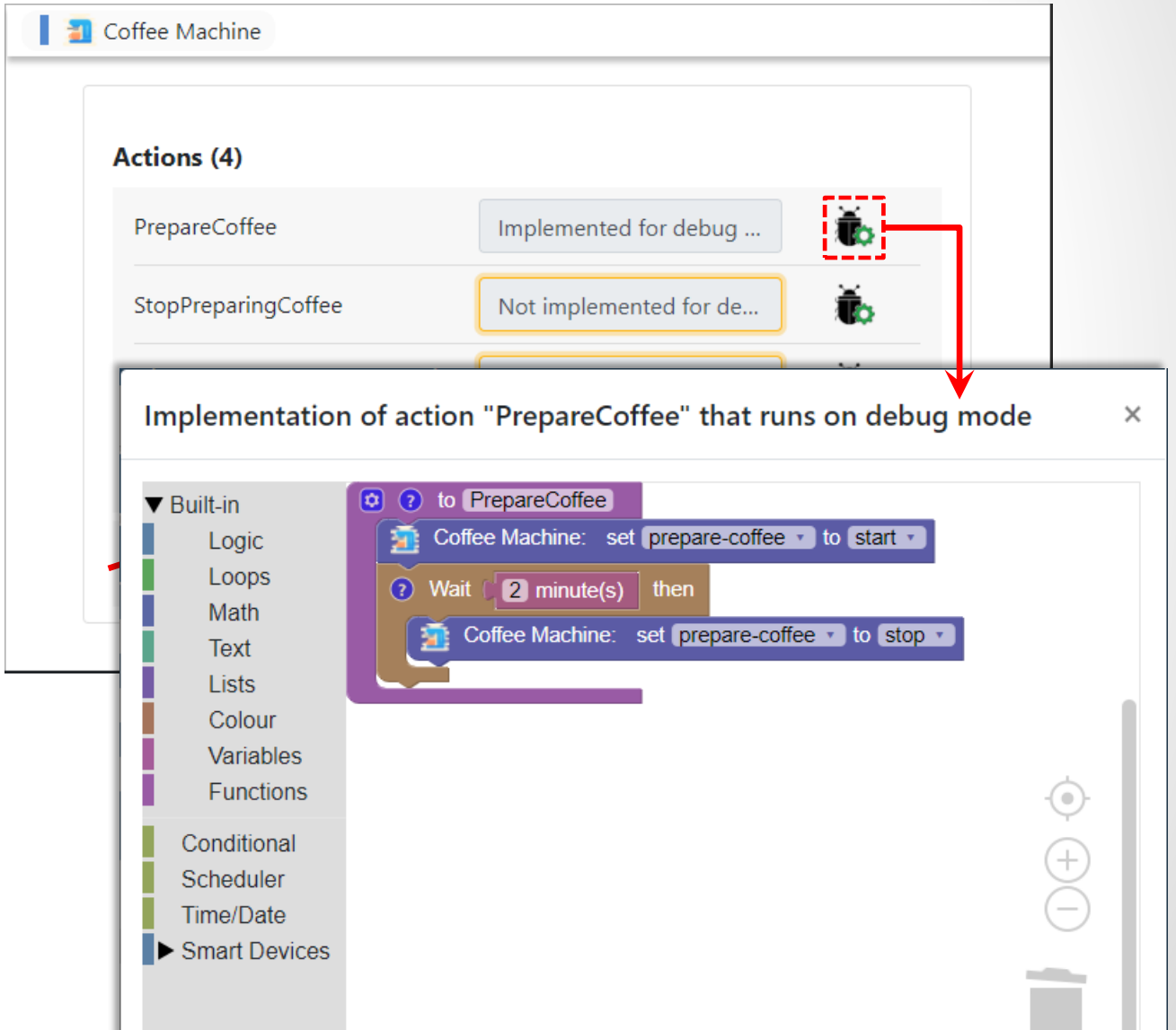
Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”
- Simulating smart environment
  - Smart devices
  - Date and time
  - Actions of smart devices



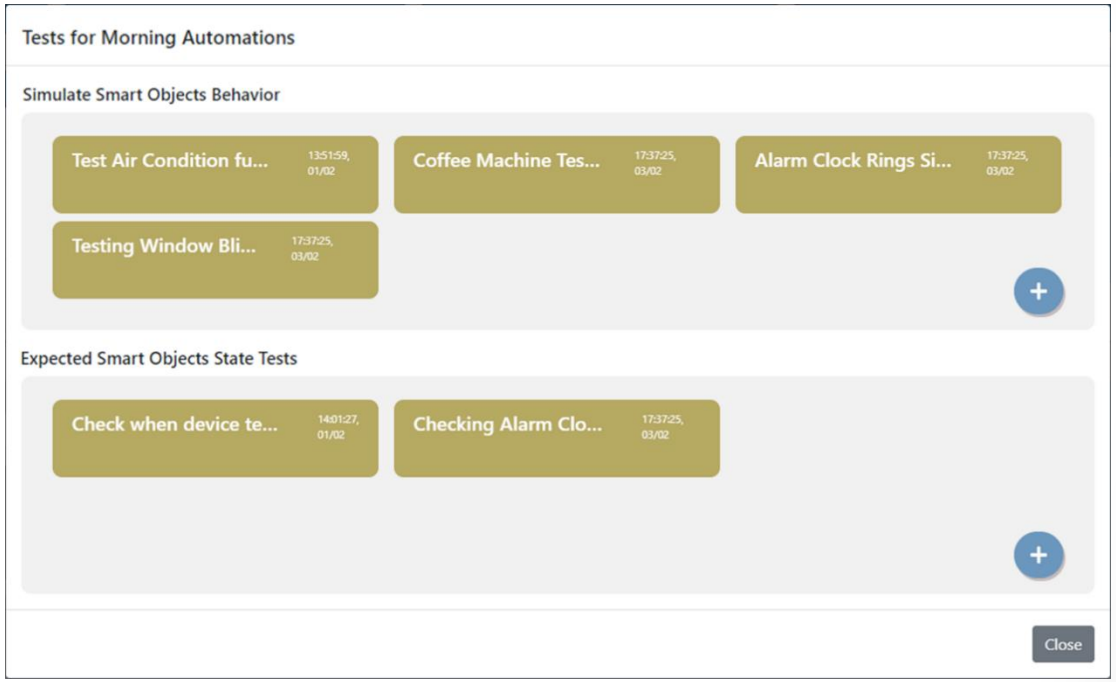
Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”
- Simulating smart environment
  - Smart devices
  - Date and time
  - Actions of smart devices



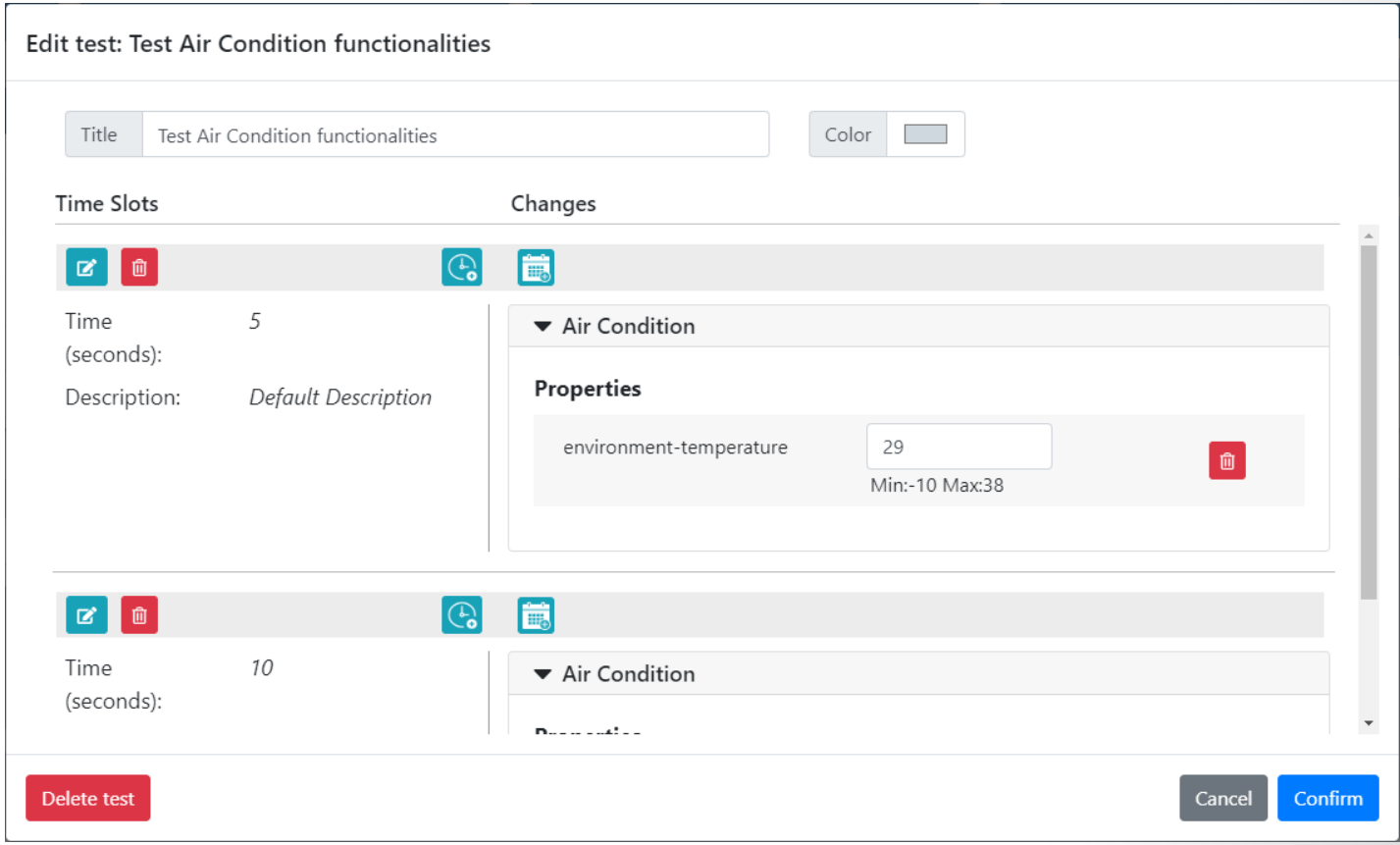
Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”
- Simulating smart environment
  - Smart devices
  - Date and time
  - Actions of smart devices



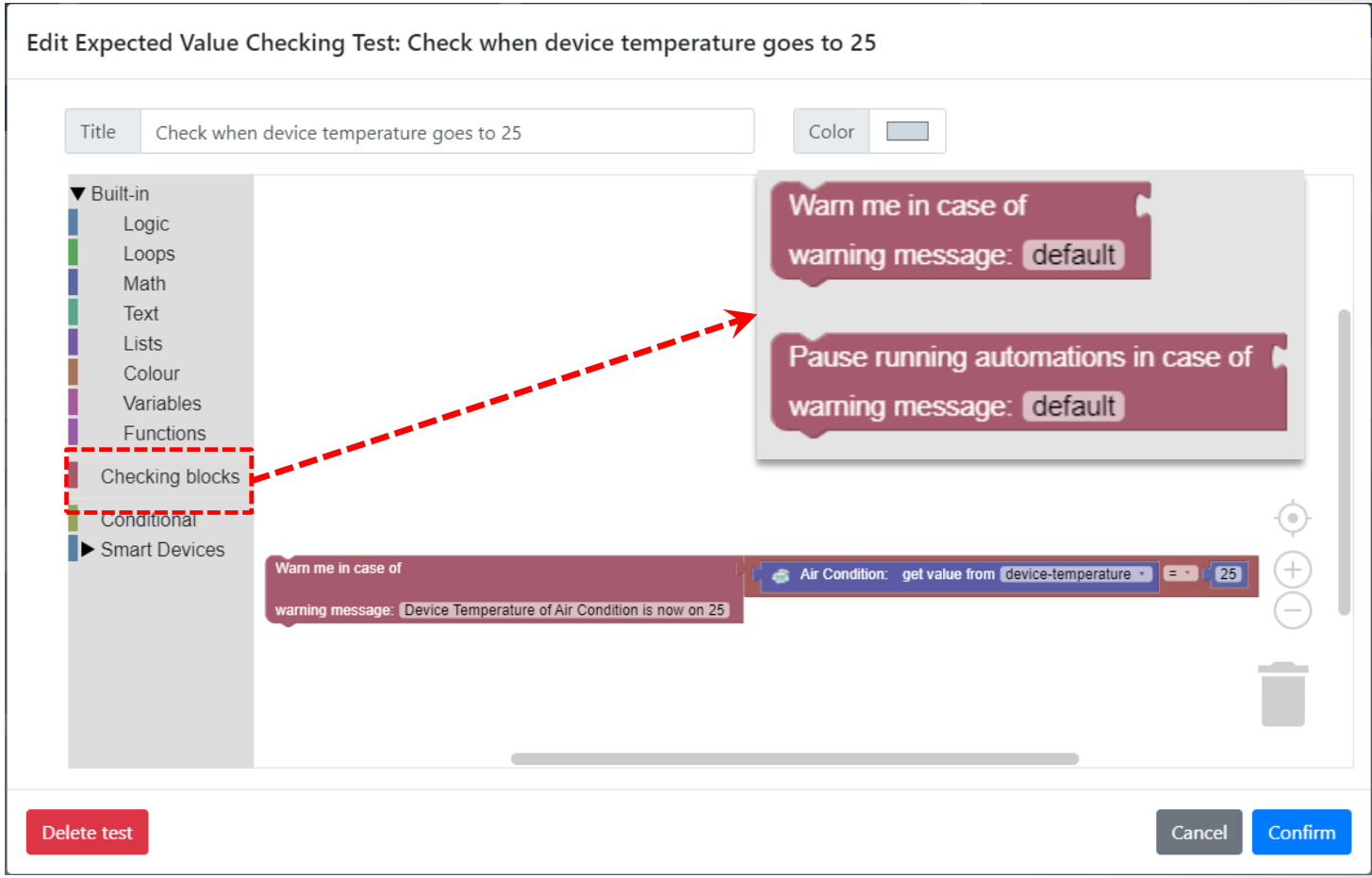
Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”
- Simulating smart environment
  - Smart devices
  - Date and time
  - Actions of smart devices
  - Behavior of smart devices



Debugging Facilities for Smart Automations

- Is debugging & testing smart automations **mission impossible**?
  - Scheduled tasks
    - “Every Tuesday, Do …”
  - Conditional tasks
    - “When smoke sensor is active then …”
- Simulating smart environment
  - Smart devices
  - Date and time
  - Actions of smart devices
  - Behavior of smart devices
  - Expected smart device values



Demo Part 2/2:

Running and Testing Automations

Conclusions

- Focusing on enable non-programmers to develop smart automations
- We developed an IDE for visual programming of smart objects in the internet of Things
  - Including custom project manager, smart device registry and grouping, device dashboard, dynamic VPL blocks handling, new VPL blocks, debugging facilities, and complete runtime

Thank you for your attention!