

Home Automation Platform Client Extension System

Suraj Sharma



Introduction

- · Low-cost system
- Aims to address the limited variety of applications currently offered by Home Automation Platforms such as Amazon Alexa
- Provides CCB API to separate applications from automation platforms and demonstrates increasing application variety via its Timer App that provides timed-control of smart plugs for the openHAB platform

	HAPCES				
Software clients] [Bridge		Device Controller	Devices
Android timer app client	Turn device 1 on	Custom API	Turn device 1 or	Home Automation	sevice 1 on Device 1
					Device 2

Background

- Limited variety of applications for Home Automation Platforms reoccurred during literature review of consumer and research platforms
- User research revealed user preferences and common platform interaction types differed, justifying the necessity of interaction software variety

Which type of input would you like to use with a home automation platform?



Chosen Platform

- openHAB was chosen based on research
- Low-cost
- Provided a REST API, allowing integration with HAPCES
- Extensive flexibility, supporting most application input and communication types
- Smart plugs were chosen due to interest in in timed-control of wired devices

OpenHAB	Add-ons					
Control	ACTIONS					
 Inbox 	Т	Tinkerforge Bine binding tinkerforge1	ding 1.12.0		INSTALL	
Configuration		Toon Binding				
Add ons		binding-toon - 2.3.0			INSTALL	
Preferences		TP-Link Smart P binding-tplinksmarth	forme Binding		UNINSTALL	
	Т	Trådfri Binding binding traditi - 2.3.0			INSTALL	
	U	UCProjects.eu R binding-usprelaytoar	telay Board Bi d - 1.12.0	nding	INSTALL	
	U	Universal Power binding-upb1 - 1.12.0	line Bus Bind	ing	INSTALL	
Paper UI	N	Velbus Binding				· .

CCB API

- Cloud-hosted CCB (Client Controller Bridge)
 REST API
- Integrates with the openHAB platform to endpoints and resources to inspect or control smart plugs
- API resource designs based on other platforms including openHAB and Samsung SmartThings



- Endpoints design supports Timer App requirements
- Endpoints were made, validated, and tested with Python
- Automatically generated interactive documentation
- Additional platforms can be supported via
 new adapter

device	8 Physical devices	
GET	/devices Returns a list of devices represented as CCB Device model JSON	
GET	/devices/{device_id} Returns a device represented as CCB Device model JSON	
feature	8 Features offered by a device	
GET	/devices/{device_id}/features Remains a device's fat of features represented as CCB DeviceFeature model JSON	
GET	/devices/{device_id}/feature_id} Riture_id Riture a device's Neture represented as CCB DeviceFeature model.350N	
states	State of a feature (can be changed via command)	
GET	/devices/{device_id}/features/{feature_id}/states/{ates_name} Remons a device's feature's state represented as CCB DeviceFeatureState model JSCN	
-	/ devices // devices // devices // devices // devices // devices // devices - men > Source a size // constant (a size // devices //	

Timer App

- Java/Kotlin Android app for openHAB
- Timed control of smart plugs via CCB API
- Customisable timer
- Link and inspect smart devices
 Consistent simple Material Design
- Consistent, simple Material Design user interface
- User accounts via Firebase Auth
- Timer cloud storage via Firebase database
- Informative error handling
- Can be tested with real or mock data



- Usability testing assessed user interaction
- Involved tasks focusing on Timer App features



- System Usability Scale assessed each user's opinion of the system
- Above average usability score (79/100)



Conclusion

- Achieved aims to successfully increase application variety for openHAB platform
- Developed low-cost interactive documentation, command-line tool, and the Timer App
- Demonstrated above average usability
- Established rudimentary foundations for future development (e.g., supporting other platforms)