iThem: Programming Internet of Things Beyond Trigger-Action Pattern

Marx Wang
Computer Science
Virginia Tech
United States
boyuan@vt.edu

Daniel Manesh
Computer Science
Virginia Tech
United States
danielmanesh@vt.edu

Ruipu Hu
Information Studies
University of Maryland, College Park
United States
rhu12@umd.edu

Sang Won Lee
Computer Science
Virginia Tech
United States
sangwonlee@vt.edu

ABSTRACT

With emerging technologies bringing Internet of Things (IoT) devices into domestic environments, trigger-action programming such as IFTTT with its simple if-this-then-that pattern provides an effective way for end-users to connect fragmented intelligent services and program their own smart home/work space automation. While the simplicity of trigger-action programming can be effective for non-programmers with its straightforward concepts and graphical user interface, it does not allow the algorithmic expressivity that a programming language has. For instance, the simple if-this-then-that structure cannot cover complex algorithms that arise from real world scenarios involving multiple conditions or keeping track of a sequence of conditions (e.g., incrementing counters, triggering one action if two conditions are both true). In this exploratory work, we take an alternative approach by creating a programmable channel between application programming interfaces (APIs), which allows programmers to preserve states and to use them to write complex algorithms. We propose iThem, which stands for intelligence of them—internet of things, that allow programmers to author any complex algorithms that can connect different IoT services and fully unleash the freedom of a general programming language. In this poster, we share the design, development, and ongoing validation progress of iThem, which piggybacks on existing programmable IoT system IFTTT, and which allows for a programmable channel that connects triggers and actions in IFTTT with versatility.

CCS CONCEPTS

• **Human-centered computing** → *User interface programming*; • **Software and its engineering** → **Application specific development environments**.

KEYWORDS

Trigger-action programming; end-user programming; Software Tools; IoT; Internet of Things; IFTTT, Do-It-Yourself; User Experience

1 INTRODUCTION

In recent years, the increasing prevalence of smart devices [10] and social media [21] has produced an overwhelming number of independent smart and connected products and services for individuals and households to manage. Trigger-action programming is a popular end-user technology that allows users to specify simple IF-THEN rules for controlling the Internet of Things and various services [1, 7]. For example, a trigger-action program involving a weather service and a smart light bulb might be: IF it starts to rain (the *trigger*), THEN change the color of my light bulb to blue (the *action*). Trigger-action programming services, such as IFTTT [17] (which stands for if-this-than-that) or Zappier[25], provide an effective way for end-users to connect fragmented services and implement their own smart home/work space automation [8, 9, 15, 22, 23].

However, many real world workflows are too complicated to be represented as simple IF-THEN algorithms. For example, the previous example of changing the light bulb color to blue when it rains does not have to run when a user is not at home. Rather, it should send a text message to the user if they are not at home. Or the algorithm should not run when the user is asleep for not distracting their sleep with blue lights. The oversimplified IF-THEN model limits users from customizing their algorithms, making it difficult to deal with growing complexities in their desired automated scenarios [15, 20, 22, 23].

Researchers have pointed out this limitation in programmable IoT. Through a series of contextual inquiries with families on home automation needs, Brich et al. have shown simple trigger-action rules are not sufficient to fulfill the needs of complex real life scenarios [2]. Researchers have looked into augmenting trigger-action programming by allowing users to combine multiple triggers and actions [15], providing additional conditional operators and constraints [9, 20] crowdsourcing algorithm authoring [16], visualizing and expanding debugging process [1, 3, 11, 26], and recommending programming recipes [4–6]. However, there remains limitation in taking advantage of the full expressivity that a programming language can afford with trigger-action-based IoT programming.

In this work, we take an alternative approach which augments the existing programmable IoT service IFTTT with a general-purpose programming language. We propose iThem—a trigger-action programming interface that can connect IFTTT triggers and actions with any user-programmed algorithm in between. iThem enables persistent state with simple database storage and retrieval, and it allows users to express complex logic in JavaScript to determine

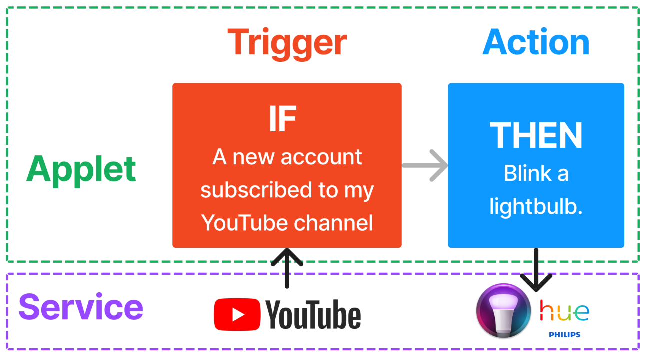


Figure 1: A applet in IFTTT where the user's desk Philips Hue light bulb will blink every time there is a new Youtube subscriber.

which actions should result from a trigger. In this poster, we will share the design pattern of the proposed system, features that enable connections between IFTTT features and any user-authored javascript code. The proposed design pattern offers a novel piggyback prototype that can abstract APIs connection by leveraging an existing IoT programming service and still allowing programmability for further personalized, automated workflows between smart devices and social media. Lastly, we will share our ongoing efforts of user study plan for validation.

2 DESIGN AND IMPLEMENTATION

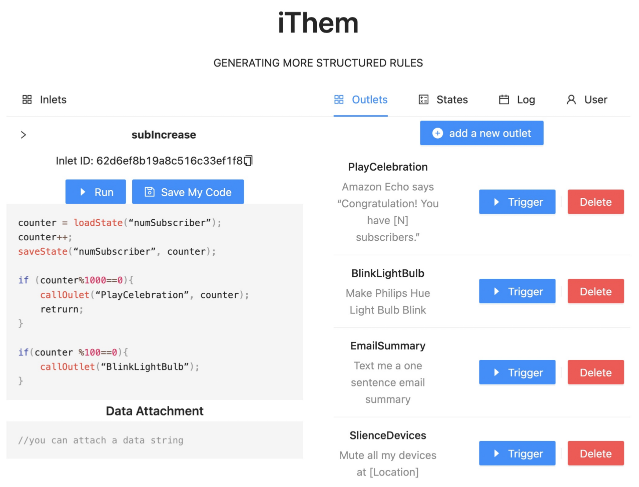
In this section, we introduce the design and features of iThem. Before we introduce iThem, we briefly introduce IFTTT concepts which are required to understand iThem's functionality. There are four different components in IFTTT: service, trigger, action, and applet.

- A *service* is a building block of IFTTT that abstracts an API of a smart device (e.g., smart thermostat, smart light bulb), social media (e.g., Twitter, Instagram), or information service (e.g., Weather). A service has a set of triggers and actions that it supports.
- A *trigger* is an event that is detected by a service, which is used to call an action on IFTTT. (e.g., Elon Musk tweeted)
- An *action* is the behavior of a service that IFTTT initiates when a trigger fires. (e.g., Text me the price of a cryptocurrency that I follow.)
- An *applet* is a specification of trigger-action relationship and represents one application in IFTTT.

Suppose a YouTube creator wants to make their Phillips Hue light bulb blink whenever there is a new subscriber. The user can create an applet that connects two services: YouTube for a trigger (a new subscriber) and Phillips Hue for an action (blink an light bulb). This applet is depicted in Figure 1.

2.1 Piggybacking on IFTTT

We take advantage of IFTTT and use its existing functions of triggers and actions that abstract communications via APIs with



```
counter = loadState("numSubscriber");
counter++;
saveState("numSubscriber", counter);

if (counter%1000==0){
    callOulet("PlayCelebration", counter);
    retrurn;
}

if(counter %100==0){
    callOutlet("BlinkLightBulb");
}
```

```
//you can attach a data string
```

PlayCelebration	Amazon Echo says "Congratulation! You have [N] subscribers."
BlinkLightBulb	Make Philips Hue Light Bulb Blink
EmailSummary	Text me a one sentence email summary
SlienceDevices	Mute all my devices at [Location]

Figure 2: iThem is an extension of IFTTT that consists of triggers, actions, and an integrated development environment. Users can connect triggers of any service to an iThem action, which is named an inlet as it takes incoming signals and can write, run, and test code in inlets (left side). An outlet is iThem's trigger in IFTTT that can be connected to other IFTTT actions. Users can call one or more outlets in the inlet via a pre-defined function callOutlet(righr side).

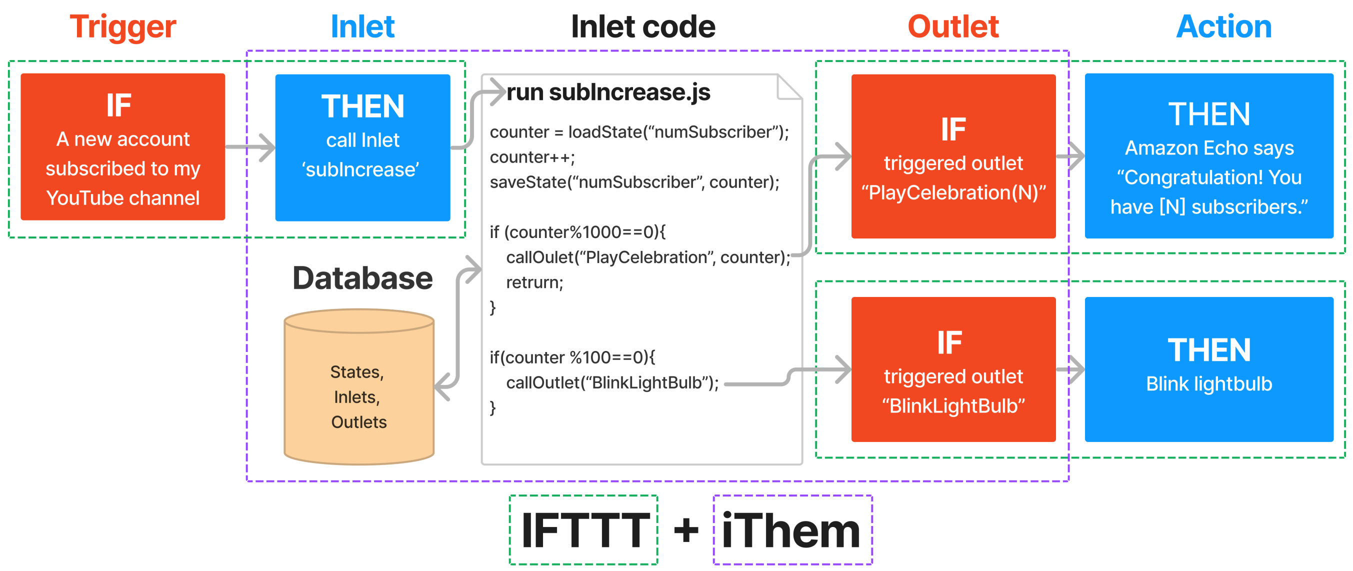
abundant number of services. This approach of developing a programmable service that can connect existing components of IFTTT is inspired from piggyback prototyping [12], which allows us to tap into an already-established, mature system.

iThem is an extension of IFTTT that consists of triggers, actions, and an integrated development environment (IDE, See Figure 2, available at https://ithem.cs.vt.edu) for programming, running, and testing. iThem serves as a programmable middle layer where user can write JavaScript code that can run by any IFTTT triggers and subsequently call any IFTTT actions. We explain three critical components of iThem: inlets, outlets, and state.

2.1.1 Inlets. To create a complex algorithm that if-this-then-that pattern cannot cover, users can split one applet into multiple applets which typically has iThem in the middle. Users can connect triggers of any service to an iThem action, which is named an *inlet* as it takes incoming signals. An inlet— iThem's IFTTT action— is always associated with one javascript file. Users can write, run, and test code in inlets, shown on the left side of interface in figure 2

2.1.2 Outlets. An outlet is iThem's trigger in IFTTT that can be connected to other IFTTT actions, shown on the right side of interface in figure 2 Users can call one or more outlets in the inlet via a pre-defined function `callOutlet()`. Then a user needs to create an applet in IFTTT, using a iThem trigger with a specific outlet handle and connect the trigger to a different action. Invoking outlets from inlets allow users to connect different applets.

2.1.3 States. A state is a backend variable to preserve a state of an algorithm across inlet calls. To unlock the full ability of programming, iThem provides the ability for users to store simple


Trigger
Inlet
Inlet code
Outlet
Action
IF
A new account subscribed to my YouTube channel
THEN
call Inlet 'subIncrease'
run subIncrease.js
Database
States, Inlets, Outlets
IF
triggered outlet "PlayCelebration(N)"
THEN
Amazon Echo says "Congratulation! You have [N] subscribers."
IF
triggered outlet "BlinkLightBulb"
THEN
Blink lightbulb
IFTTT + iThem

```
counter = loadState(“numSubscriber”);
counter++;
saveState(“numSubscriber”, counter);

if (counter%1000==0){
   callOulet(“PlayCelebration”, counter);
   retrurn;
}

if(counter %100==0){
   callOutlet(“BlinkLightBulb”);
}
```

Figure 3: A use case of iThem and IFTTT for John and his cat Youtube Channel: With iThem, John was able to create a complex automation where his desk light only blinks when the number of new subscribers reaches a multiple of 100 and his Amazon Echo proudly announces the new subscriber count if the new subscribers number is at the multiple of 1000. John used one incoming applet to call inlet 'subIncrease' if there's a new subscriber, and two applets that will be called through 'PlayCelebration' and 'BlinkLightBulb' outlets to perform light-bulb blinking and Amazon Echo announcement action. John stored the number of subscribers in a state variable 'numSubscriber' and wrote his algorithm in 'subIncrease.js'

or complex data, preserving any state they may have. Users can create a variable from the interface and can programmatically read and write data into variables via pre-defined `loadState()` and `saveState()`.

2.2 Implementation

iThem platform is implemented with NextJS and React as frameworks and MongoDB as a flexible database. The communication between iThem and IFTTT is implemented based on IFTTT API documentation. The code-running environment in inlets utilizes 'vm2'[1], a Node sandbox module that allows users to run untrusted code securely in a single process. IFTTT services and iThem allow passing a simple string data field from trigger to action and inlets allow users to read this user-customizable data.

2.3 Example iThem scenario

John is a software developer and he created a youtube channel to showcase his cute cat. Initially, John's cat's channel only gained a few followers per week. John felt very proud nonetheless and wanted to make an automation that can make his desk Philips Hue light bulb blink every time there is a new subscriber. He created an applet in IFTTT to do precisely that, as depicted in Figure 1.

As John uploaded more and more funny and cute videos of his cat, his cat channel started to gain popularity. Now his desk light would blink a couple times every hour, sometimes even several times per minute. His IFTTT applet was now an annoyance. He wanted to set a condition that the desk light only blinks when the number of new subscribers reaches a multiple of 100. However, he cannot do that with IFTTT because they do not have such a trigger and he cannot keep track of subscriber numbers.

With iThem, however, he was able to create his ideal automation. He created a variable 'numSubscriber' in iThem to store the total number of subscribers. In IFTTT, he created an applet with the same IF condition (i.e. IF a new account subscribed to my cat channel) but with a different THEN action (i.e. THEN call my program in iThem). He also created an outlet called "BlinkLightBulb" connecting to an applet in IFTTT with a action 'blink lightbulb'. In addition to make the desk light blink, he also want to have his Amazon Echo proudly announce the new subscriber count if the new subscribers number is at the multiple of 1000, so he created another outlet in iThem, along with the corresponding applet in IFTTT. Figure 3 shows the overall structures of IFTTT and iThem of this automation and the code.

3 ONGOING EFFORTS AND FUTURE WORK

We plan to validate our system with a two-part user study. In the first part of the study, we will introduce iThem to participants and then ask them to solve a series of problems using the system. This part of the study is mainly to evaluate and collect feedback on the user interface. We will evaluate iThem based on how well participants were able to complete the tasks or not, as well as through a usability survey and exit interview.

For the second part of the study, the same participants will be given a week to come up with their own algorithms to implement

[1]https://github.com/patriksimek/vm2

using iThem. At the end of the week, we will have a short interview to probe for any shortcomings with iThem's computational model and also to gauge participants' attitudes about using a system like iThem in the future.

While our planned study is aimed at validating our system with participants who have coding experience, iThem could also provide a valuable space for non-programmers to learn computational thinking skills. One future direction is to explore how we can leverage iThem to nudge general users to learn programming through the Do-It-Yourself style smart home configuration projects [24]. Constructionism-based learning, or project-based learning[13], is a popular and widely adopted pedagogical strategy in computer science education [18] and research has shown that projects with a combination of physical devices and programming can result in greater engagement and understanding in programming concepts[14, 19]. Through developing their own work/home automation services, iThem will give end-users novel motivations to learn to program with something that are relatable to their lives and be better adapt in the ever-changing technology world.

REFERENCES

[1] Will Brackenbury, Abhimanyu Deora, Jillian Ritchey, Jason Vallee, Weijia He, Guan Wang, Michael L. Littman, and Blase Ur. 2019. How Users Interpret Bugs in Trigger-Action Programming. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI: http://dx.doi.org/10.1145/3290605.3300782

[2] Julia Brich, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub. 2017. Exploring End User Programming Needs in Home Automation. *ACM Trans. Comput.-Hum. Interact.* 24, 2, Article 11 (apr 2017), 35 pages. DOI: http://dx.doi.org/10.1145/3057858

[3] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019a. Empowering End Users in Debugging Trigger-Action Rules. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. DOI: http://dx.doi.org/10.1145/3290605.3300618

[4] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019b. EUDoptimizer: Assisting End Users in Composing IF-THEN Rules Through Optimization. *IEEE Access* 7 (2019), 37950–37960. DOI: http://dx.doi.org/10.1109/ACCESS.2019.2905619

[5] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019c. RecRules: Recommending IF-THEN Rules for End-User Development. *ACM Trans. Intell. Syst. Technol.* 10, 5, Article 58 (sep 2019), 27 pages. DOI: http://dx.doi.org/10.1145/3344211

[6] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2020. TAPrec: Supporting the Composition of Trigger-Action Rules through Dynamic Recommendations. In *Proceedings of the 25th International Conference on Intelligent User Interfaces (IUI '20)*. Association for Computing Machinery, New York, NY, USA, 579–588. DOI: http://dx.doi.org/10.1145/3377325.3377499

[7] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2022. How do end-users program the Internet of Things? *Behaviour & Information Technology* 0, 0 (2022), 1–23. DOI: http://dx.doi.org/10.1080/0144929X.2022.2071169

[8] Luigi De Russis and Fulvio Corno. 2015. HomeRules: A Tangible End-User Programming Interface for Smart Homes. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. Association for Computing Machinery, New York, NY, USA, 2109–2114. DOI: http://dx.doi.org/10.1145/2702613.2732795

[9] Giuseppe Desolda, Carmelo Ardito, and Maristella Matera. 2017. Empowering end users to customize their smart environments: model, composition paradigms, and domain-specific tools. *ACM Transactions on Computer-Human Interaction (TOCHI)* 24, 2 (2017), 1–52.

[10] Colin Dixon, Ratul Mahajan, Sharad Agarwal, A.J. Brush, Bongshin Lee, Stefan Saroiu, and Paramvir Bahl. 2012. An Operating System for the Home. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. USENIX Association, San Jose, CA, 337–352. https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/dixon

[11] Giuseppe Ghiani, Marco Manca, Fabio Paternò, and Carmen Santoro. 2017. Personalization of Context-Dependent Applications Through Trigger-Action Rules. *ACM Trans. Comput.-Hum. Interact.* 24, 2, Article 14 (apr 2017), 33 pages. DOI: http://dx.doi.org/10.1145/3057861

[12] Catherine Grevet and Eric Gilbert. 2015. Piggyback Prototyping: Using Existing, Large-Scale Social Computing Systems to Prototype New Ones. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 4047–4056. DOI: http://dx.doi.org/10.1145/2702123.2702395

[13] H.M. Havenga. 2015. Project-based learning in higher education : exploring programming students' development towards self-directedness. *South African Journal of Higher Education* 29, 4 (2015), 135–157. DOI: http://dx.doi.org/10.10520/EJC182452

[14] Ting-Chia Hsu, Shao-Chen Chang, and Yu-Ting Hung. 2018. How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education* 126 (2018), 296–310. DOI: http://dx.doi.org/https://doi.org/10.1016/j.compedu.2018.07.004

[15] Justin Huang and Maya Cakmak. 2015. Supporting Mental Model Accuracy in Trigger-Action Programming. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 215–225. DOI: http://dx.doi.org/10.1145/2750858.2805830

[16] Ting-Hao K. Huang, Amos Azaria, Oscar J. Romero, and Jeffrey P. Bigham. 2019. InstructableCrowd: Creating IF-THEN Rules for Smartphones via Conversations with the Crowd. *Human Computation* 6, 1 (Sep. 2019), 113–146. DOI: http://dx.doi.org/10.15346/hc.v6i1.7

[17] IFTTT. 2022. WTF IS IFTTT? (2022). https://ifttt.com/explore/new_to_ifttt

[18] Dimitra Kokotsaki, Victoria Menzies, and Andy Wiggins. 2016. Project-based learning: A review of the literature. *Improving Schools* 19, 3 (2016), 267–277. DOI: http://dx.doi.org/10.1177/1365480216659733

[19] Utku Köse. 2010. A web based system for project-based learning activities in "web design and programming" course. *Procedia - Social and Behavioral Sciences* 2, 2 (2010), 1174–1184. DOI: http://dx.doi.org/https://doi.org/10.1016/j.sbspro.2010.03.168 Innovation and Creativity in Education.

[20] Sarah Mennicken, Jo Vermeulen, and Elaine M. Huang. 2014. From Today's Augmented Houses to Tomorrow's Smart Homes: New Directions for Home Automation Research. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. Association for Computing Machinery, New York, NY, USA, 105–115. DOI: http://dx.doi.org/10.1145/2632048.2636076

[21] Manya Sleeper, William Melicher, Hana Habib, Lujo Bauer, Lorrie Faith Cranor, and Michelle L Mazurek. 2016. Sharing personal content online: Exploring channel choice and multi-channel behaviors. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. 101–112.

[22] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. 2014. Practical Trigger-Action Programming in the Smart Home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 803–812. DOI: http://dx.doi.org/10.1145/2556288.2557420

[23] Blase Ur, Melwyn Pak Yong Ho, Stephen Brawner, Jiyun Lee, Sarah Mennicken, Noah Picard, Diane Schulze, and Michael L. Littman. 2016. Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. Association for Computing Machinery, New York, NY, USA, 3227–3231. DOI: http://dx.doi.org/10.1145/2858036.2858556

[24] Jong-bum Woo and Youn-kyung Lim. 2015. User Experience in Do-It-Yourself-Style Smart Homes. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 779–790. DOI: http://dx.doi.org/10.1145/2750858.2806063

[25] Zapier. 2022. Learn key concepts in Zapier. (Feb 2022). https://zapier.com/help/create/basics/learn-key-concepts-in-zapier

[26] Valerie Zhao, Lefan Zhang, Bo Wang, Shan Lu, and Blase Ur. 2020. Visualizing Differences to Improve End-User Understanding of Trigger-Action Programs. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–10. DOI: http://dx.doi.org/10.1145/3334480.3382940