USER STUDY 1

Role : WAREHOUSE ROBOT OPERATOR

Scenario : You are a warehouse worker responsible for operating a robot using our DSL (Domain Specific Language). You have access to a number of instructions/ commands that will allow you to move the robot around, transport items and order new items to be delivered.

List of Commands :

- goTo(*location*)
    - Orders the robot to go to the specified *location (*option: warehouse or fronthouse)
- if / if(!)
    - *if* something happens do one thing; *if not* something happens, do the following
- pickUpItem()
    - Order the robot to pick up an item
- dropOffItem()
    - Order the robot to drop off an item
- itemIsAvailable()
    - Check if there are items available
- restockItem(type, qty, order)
    - Restock items
- repeatUntilDone(*action*)
    - Repeat the *action* until it's not necessary anymore

After the second meeting with TA, we modified our prototype by adding more features and variables.

1. **Customer order can be created with multiple items and quantities.**
2. **Define shelves as item locations, customers can create shelves by one shelf per item rule.**
3. **For function restockItem, accepting more parameters.**

- **The modified prototype, new and old version have been updated to github repo.**
- **Two user studies are done, one with the old prototype and one with the new prototype, Please refer to the *User Study 1 file*.**

New version:

Updated Example:

1. Restock 5 item A
   restockItem(A,5)
2. Create an order containing 2 item A
   customerOrder = [{A, 2}]
3. You can get customerOrder's and product's information using " . ", like:
    - "customerOrder.products" to get all the products in your order, so you can generate a loop like "every product in customerOrder.products"
    - for each product "product.location" for locations, "product.amount" for amount.

Updated Task:

1. Check if item C is available and if not, restock 10 item C.
2. Create an order containing 5 item A, 2 item B and 4 item C.
3. After creating the order, grab all the products in the order created.

Standard Solution:

For task 1:

```
// Runs if warehouse is empty
if (!ItemIsAvailable){              // if the order is not available in the warehouse we need to restock
        restockItem(C, 10)     // order items for restock
}
```

For task 2:

```
customerOrder = [ {A,5}, {B,2}, {C,4} ]    // what the customer wants
```

For task 3:

```
// Always runs
every product in customerOrder.products {
        goTo(product.location) // goTo(warehouse) // Do we need separate shelves for each item?
        pickUp(product.amount)
        goTo(fronthouse)
        dropOff(product)
}
```

User 2's Feedback :

- The first two tasks are easy because there are examples provided, but the third task took the participant some time to figure out.
- Confused if white space is necessary or not in each command (the tested user wants to get everything right to prevent potential mistakes, which also took some time)
- In general, the commands are easy to learn and easy to use, but a "Help", or "example" box would be helpful so that they can have a reference and know all the grammar rules when entering the command.
- Love the name given to all the commands because she can figure out what each command means simply be their name given.
- Similar to natural language, but all the brackets are kinda "annoying" for the first time typing it. Once the user gets used to it, it may be fine.
- Would be better if there are some instructions to state that "shelf 1 only contains item A", and the user is wondering what if there is more item A than the shelf can contain, can users have the ability to assign another shelf.

User 2's Ideas for a Variable Syntax (ItemOrder) :

- Do not like the brackets, but if the user gets used to it, they are acceptable.
- If the " . " can be simplified or consistent. Having product.amount, product.location, and customerOrder.products together feels like she can . everything to get some information out of it.

Old version:

Old Task :

1. Example: Assume item is in stock, move item from Warehouse to Front of house.
2. Example: Assume item is not on the shelf in warehouse, order item for restock and repeat MAX_NUMBER_ITEMS.
3. User Input: Check inventory, then Combine situation 1 and 2 into one command.

User 1's Feedback :

- Commands are intuitive and was relatively easy to figure out what to do.
- Liked the natural language instead of using programming syntax.
- Would like to have white space for structure so it doesn't look like a continuous blob.
- Wouldn't mind some symbols accompanying the if / ifNot conditional blocks.

User 1's Ideas for a Variable Syntax (ItemOrder) :

- Myorder = 3D,110C,750G
- create anOrder of 3D, 110C, 760G
- create itemOrderB ofType B = 10

Sample Solution :

Task 1:

goTo(warehouse);
pickUpItem;
goTo(fronthouse);
dropOffItem;

Task 2:

goTo(warehouse);
itemIsAvailable;
restockItem;
pickUpItem;
goTo(frontHouse);
dropOffItem;

Task 3:

goTo(warehouse);
if, itemIsAvailable, pickUpItem, ifNot, repeatUntilDone(restockItem);
pickUpItem;
goTo(fronthouse);

dropOffItem;

Conclusion for 2 user studies:

1. In general, the language is easy to learn and use. All the commands are intuitive, and 2 tested users can understand the commands using the name of the function and the explanation provided.
2. Some grammar can be simplified, like creating orders
    - Currently is defined as: “customerOrder = [{A,5}, {B,2}, {C,4}] ”, but tested user 2 did not like the brackets, and tested user 1 suggested can be simplified to “create anOrder of 3D, 110C, 760G”, so 2 users’ suggestions can be combined.
3. A “help” box indicating all the grammar, syntax and examples can be useful.
4. User suggested we enforce some sort of white space structure for readability