Saturday 27.11.1441
18.7.2020

Developing Course — challeuge.07 budget (Abstract)
insert date inside UI

budget Controller.

"Privacy-module"

id
description
value

- Construction function-"ExPense"
- Construction function-"InCome"
- object-data"

- allItems — array-exP.
  — array-inC.
- total — No.-exp.
  — No.-inC.
- budget: No.
- PerCentage: No.

method-CalClateTotal(type) important

```
var Sum =0;
data.allItems[type].forEach(function(Current){
Sum += Current.value;
data.total[type] = Sum;
```

"Public-module"

method-addItem(type,des,val)

```
var ID , newItem;
ID = Last ID + 1;
If type = "exp"
        = "inC"   "insert inside Construction
                   Function"
data.allItem[type].Push(newItem);
  return newItem;
```

method-Calculate Budget

```
CalCulate Total ("exp");
CalCulate Total ("inC");

data.budget = inC. - exP.

data.PerCentage = ((exP / inC.) *100)
                  Math.round
```

UI Controller,

Privacy-module

object-domString

- type data:
- descriptiondata:
- value data:
- btndata:
- inComeContiner:
- expeseContiner:
- budget label:
- inCome label:
- expese label:
- PerCentage:

Public module

object-getInputData important

```
return{
type: DoM.value,
description: DoM.value,
value: parseFloat(DoM.value)
```

method-getDomstring

```
return domString;
```

method-add List Item(type,obj) important

```
var html, element, newhtml;
If type = "exp"  → html  %id%
   type = 'inc'          %des%
                         %value%
                 → element  Dom
newhtml = html.replace(%id%, obj.id);
   "    =  "  .  "  (%des%, obj.des);
   "    =  "  .  "  (%value, obj.value);
DoM(element).
insertAdjacentHTML
      ("beforeend", newhtml);
```

Controller.

Privacy-module

method-updateBudget

```
budgetctrl.CalculateBudget();
var budget = uiCtrl.getBudget();
uictrl.displayBudget(budget);
```

important

method-ctrlAddItem

```
var input, newItem;
  input = uiCtrl.getinputData();

IF(
 input.description
     !== ""  &&
 input.value >0 &&
 !isNaN(input-value)
     ){
var newItem
= budgetctrl.addItem(input.type,
                     input.de
                     input.value);
→ uictrl.addListItem(inpute.type,
                     newItem);
→ uictrl.clearfields;
→ update Budget();
}
```

budget Controller

object - getBudget

```
return {
budget: data.budget,
totalInc: data.total.inc,
total exp: data.total.exp,
Percentage: data.Percentage
```

Testing:

```
Console.log(data);
```

UI Controller

method - clearFields — important

```
var fields, fieldArr;

Fields = DoM.querySelectorAll(description + ", " + value)

FieldArr = Array.Prototype.slice.
        Call(fields);

FieldArr.ForEach(function(current){
  current.value = "";
  });
```

method - displayBudget (obj) — important

```
DoM(budget lebel).textContent
                 = obj.budget;

DoM(incomelebel).TextContent
                 = obj.totalInc;

DoM(expense lebel).TextContent
                 = obj.totalexp;

DoM(Percentage).TextContent
                 = obj.PerCentage;
```

```
obj.Percentage    IF     else
     > 0
obj.Percentage + "%"     "----"
```

Controller

method - SetupEventListener

```
var dom = uictrl.domstring();
addEventListener("click, -);
addEventListener('Keypress, -);
```

Public module

method - intial

```
Console.log("Application has
             started");
```

```
uictrl.displayBudget({
      budget: 0,
  totalInc: 0,
  Total exp: 0,
  Percentage: 0
        });

SetupEventListener();
```

```
Controller.intial();
```