

Budget#

O jeito inteligente de organizar suas finanças pessoais

LUCA CARUSO

Lógica da Computação - Insper 2025.1

Inspiração



01

Objetivo: Trocar de Celular

No meio deste semestre, meu celular começou a dar sinais que seu fim de aproximava. Com isso, percebi que precisaria organizar as minhas finanças para guardar um dinheiro para comprar um celular novo.



02

Problema: Planilhas são chatas

Fazer uma planilha que fosse eficiente, fácil de usar demandaria muito tempo, o que me faltava em meio aos projetos do semestre. Além disso, ninguém gosta de ficar fazendo planilha



03

Solução: Budget#

Diante da proposta da APS e do problema que estava querendo superar, desenvolvi a Budget# com o intuito de poder fazer um balanço das minhas receitas e despesas, apenas fazendo um “log” do que eu gastasse ao longo do mês e compilando isso em uma visualização user-friendly.

Características

Bloco de setup

Dentro dele você configura seus **orçamentos e receitas e variáveis**. Estes parâmetros serão utilizados durante o evaluate para **calcular o balanço total, orçamentos estourados ou servir como base de cálculo**.

Todo código B# deve ter o bloco de setup

Variáveis e Operações

Você pode declarar **variáveis para serem usadas dentro ou fora do setup**. Como por exemplo declarar um valor de 20 reais na variável UberDaily, que poderia ser usada no **calculo do budget** (set budget transport to UberDaily * 30) ou no **cálculo do gasto** (spend dailyUber on transport every day for 30 days)

Loops

Durante o log de despesas, você pode fazer **loops de gastos**. Por exemplo, como já sei que usarei o uber 20 dias no mês ao invés de declara um gasto 20 vezes eu posso simplesmente declarar: **spend dailyUber on transport every day for 20 days**

Log de Despesas

Ao invés adicionar cada despesa realizada à sua respectiva categoria em uma planilha excel, usando o Budget# você pode **declarar de forma mais natural, indicando apenas o valor e a categoria**, deixando a rotulação o compilador. Ex: spend 100 on food

Condicionais

Os condicionais são implícitos ao código, de forma que eles acontecem durante a compilação, quando **são verificados se houve ou não estouro do budget estabelecido, criando um flag que será mostrado no relatório gerado**

Relatório

Ao termino da avaliação da AST, o código **compila todas as informações em um relatório html**, mostrando de forma clara **o balanço do mês e das categorias de orçamento**.

Todo código deve ter a chamada da geração do relatório ao final report month 06/2025

Exemplo de Código

```
JUN25.BUDGET

# Mês de controle: Junho de 2025
for month 06/2025

setup {
  var dailyUber    = 20
  var daysInMonth = 30
  set budget transport to dailyUber * daysInMonth
  set budget party to 1000
  set budget food to 300 + 200
  add income salary 4000 - 500
  add income bonus 1000
}

# Despesas
spend dailyUber on transport every day for daysInMonth days
spend 650 on transport
spend 100 on food
spend 3000 on party
spend 100 on party

# Relatório
report month 06/2025
```



Expense Report for 06/2025			
Category	Spent	Budget	Status
food	100	500	OK
party	3100	1000	OVER by 2100
transport	1250	600	OVER by 650

Summary

Total Income: 4500
Total Expense: 4450
Net Balance: 50

Alerts

party exceeded budget by 2100

transport exceeded budget by 650

EBNF

<PROGRAM> ::= <MONTH-DECL> <SETUP-BLOCK> { <STATEMENT-BODY> } <REPORT-STMT>;

<MONTH-DECL> ::= "FOR" "MONTH" <NUMBER> "/" <NUMBER>;

<SETUP-BLOCK> ::= "SETUP" "{" { <VAR-STMT> | <BUDGET-STMT> | <INCOME-STMT> } "}";

<STATEMENT-BODY> ::= <VAR-STMT> | <EXPENSE-STMT>;

<VAR-STMT> ::= "VAR" <IDENTIFIER> "=" <EXPR>;

<BUDGET-STMT> ::= "SET" "BUDGET" <IDENTIFIER> "TO" <EXPR>;

<INCOME-STMT> ::= "ADD" "INCOME" <IDENTIFIER> <EXPR>;

<EXPENSE-STMT> ::= "SPEND" <EXPR> "ON" <IDENTIFIER> [<RECUR-CLAUSE>];

<RECUR-CLAUSE> ::= "EVERY" "DAY" "FOR" <EXPR> "DAYS";

<REPORT-STMT> ::= "REPORT" "MONTH" <NUMBER> "/" <NUMBER>;

/* EXPRESSÕES ARITMÉTICAS */

<EXPR> ::= <TERM> { ("+" | "-") <TERM> };

<TERM> ::= <FACTOR> { ("*" | "/") <FACTOR> };

<FACTOR> ::= <NUMBER> | <IDENTIFIER> | "(" <EXPR> ")";

/* IDENTIFICADORES */

<IDENTIFIER> ::= <LETTER> { <LETTER> | <DIGIT> | "_" };

/* LITERAIS NUMÉRICOS */

<NUMBER> ::= <DIGIT> { <DIGIT> };

/* CARACTERES - APENAS PARA DEFINIÇÃO */

<DIGIT> ::= "0" ... "9";

<LETTER> ::= "A" ... "Z" | "a" ... "z";

<ANY-CHAR> ::= <LETTER> | <DIGIT> | <SYMBOL> | <WHITESPACE>;

<SYMBOL> ::= "(" | ")" | "{" | "}" | "-" | "+" | "*" | "/" | "=" | "\"";

<WHITESPACE> ::= " " | "\t";

<NL> ::= "\n";