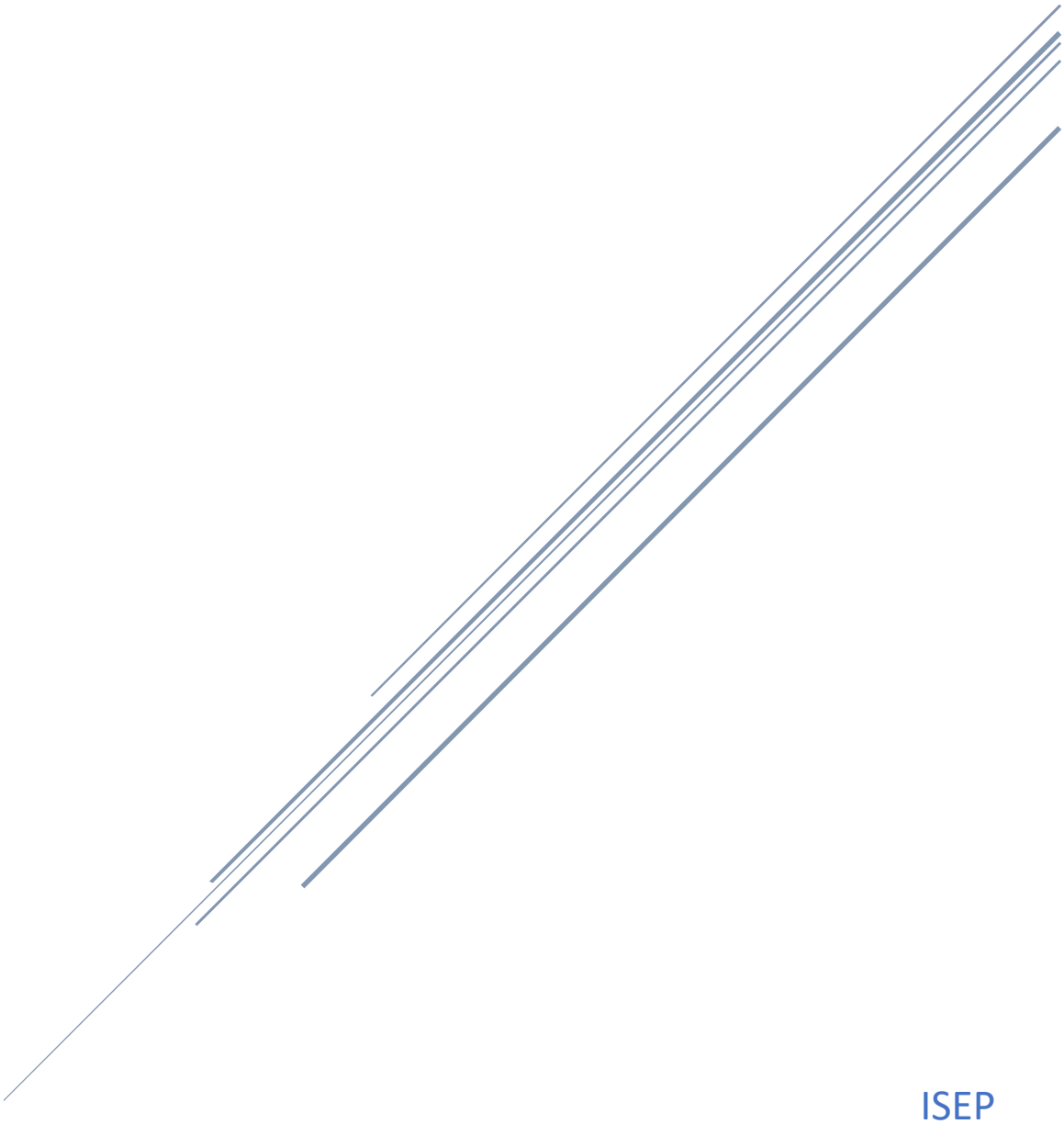
Valida Script

Gramática Grupo 4 2DL



ISEP
Engenharia Informática

Índice

Um dos requisitos apresentados pelo cliente foi o desenvolvimento de uma linguagem/gramática de suporte ao sistema para expressar, entre outras coisas, validações de formulários e atividades automáticas.

Para tal, o grupo desenvolveu a gramática "Valida Script", que têm como objetivo principal executar validações condicionais e aritméticas. Para tal a nossa gramática foi dividida em dois pontos principais.

```
grammar ValidaScript;

prog: stat*  | validaExp*;
stat: operacoesAritmeticas NEWLINE # printExpr
| ID '=' operacoesAritmeticas NEWLINE # assign
| NEWLINE # blank
```

Operações Aritméticas

Para fazer a validação de operações aritméticas, bem como disponibilizar a capacidade de substituir valores pré-definidos em emails, o cálculo de promoções, e a identificação de um produto ou cliente através de um identificador único (ID) a nossa gramática oferece suporte no que toca a verificação de operações aritméticas.

De momento, a linguagem suporta as operações básicas de adição, subtração, multiplicação e divisão, sendo que será possível acrescentar operações mais complexas, caso o cliente necessite da inclusão das mesmas.

Exemplo de funcionamento:

Verificação de um cliente/produto através de ID:

O utilizador deve utilizar um ID (1 ou mais caracteres alfabéticos minúsculos), seguido do caracter "=" e posteriormente o ID que deseja verificar.

Estruturas Condicionais

De modo a ajudar a implementar a validação de operações aritméticas bem como proceder a validações de cariz mais geral, foi desenvolvido este módulo.

As estruturas condicionais ajudam a fazer a validação de scripts mais completos assim como de aproveitar validações aritméticas para proceder a determinados procedimentos.

Para tal, foram desenvolvidas várias verificações de modo que um processo possa ser automatizado, adaptando-se a diferentes cenários que possam surgir.

Exemplo de funcionamento:

Um colaborador pretende marcar as suas férias.

Se o número de dias escolhido pelo colaborador for 0:

- O colaborador terá que escolher pelo menos um dia.

Se o número de dias ultrapassar o seu banco de dias de férias:

- O colaborador deve escolher apenas o número de dias a que têm direito.

Se os dias escolhidos pelo colaborador já estiverem reservados por outro colaborador:

- O colaborador deve escolher dias livres

Sessão de esclarecimento

A equipa está disponível para proceder a sessões de esclarecimento/ formação relativas à gramática apresentada neste documento.

Contacto

- Grupo4_2DL@gmail.com

Anexos

```
grammar ValidaScript;

prog: stat*  | validaExp*;
stat: expr NEWLINE # printExpr
| ID '=' expr NEWLINE # assign
| NEWLINE # blank

;
expr: expr op=('*'|'/') expr # MulDiv
| expr op=('+'|'-') expr# AddSub
| INT # inte
| ID # id
| '(' expr ')' # parens
;

validaExp: validaExp DEFINE exprExp EXPREGULAR     #defineExpRegular
      | SE exprExp VAZIO ENTAO exprExp NAOVAZIO       #validaEntreCampos
      | SE exprExp NAOVAZIO ENTAO exprExp VAZIO       #validaEntreCampos
      | SE exprExp MAIOR exprExp ENTAO exprExp VAZIO      #validaEntreCampos
      | SE exprExp MAIOR exprExp ENTAO exprExp NAOVAZIO   #validaEntreCampos
      | SE exprExp MENOR exprExp ENTAO exprExp NAOVAZIO   #validaEntreCampos
      | SE exprExp ATR exprExp ENTAO exprExp NAOVAZIO   #validaEntreCampos
      | DEFINE exprExp NAOVAZIO                   #defineCampoNaoVazio
      | DEFINE exprExp VAZIO                      #defineCampoPodeSerVazio
;

exprExp:    INT          #int
        | NOMEATRIBUTO   #nomeAtributo
        | MAIOR          #maior
        | MENOR          #menor
        | ATR            #igual
;

NEWLINE : [\r\n]+ ;
INT:[0-9]+;
ID:[a-z]+;
MUL : '*' ; // assigns token name to '*' used above in grammar
DIV : '/' ;
ADD : '+' ;
SUB : '-' ;
ATR : '=' ;
LPR : '(' ;
```