

## << abstract >> Function Strategy

- arguments: Map<String, Value>
- is-valid-input: Predicate<Value>
- output: ValueObserver

```
+ << final >> updateOutput(): void
# << abstract >> result: String
# getType(): Type << abstract >>
# add Value(Spreadsheet, String, String): void
# add Range(Spreadsheet, String, String): void
# get Value(String): (void) Value
# get Range(String): Collection<Value>
+ Clean Function(): void
```

## << abstract >> Binary Function

```
# << final >> result(): String
# << final >> getType: Type
# << abstract >> compute(int, int): int
```

## << abstract >> Range Function

```
# << final >> result(): String
# << abstract >> compute(Collection<Values>) String
```

## AVERAGE

```
# compute(Collection<Values>): String
```

## PRODUCT

```
# compute(Collection<Values>): String
```

## CONCAT

```
# compute(Collection<Values>): String
```

## COALESCE

```
# compute(Collection<Values>): String
```

## Reference Call

```
# << final >> result(): String
# << final >> getType(): Type
```

## ADD

```
# compute(int, int): int
```

## SUB

```
# compute(int, int): int
```

## MUL

```
# compute(int, int): int
```

## DIV

```
# compute(int, int): int
```

xxl. functions

## Value

(Type)

- \_type: Type
- \_value: String
- function: FunctionStrategy
- observers: Collection <Observer>  
ValueObserver

- + update Expression (String literal): void
- + update Expression (Spreadsheet, String, String...): void
- + set Error(): void
- + set Value (String, Type): void
- + getType(): Type
- + update Value(): void
- + attach (Value Observer): void
- + detach (Value Observer): void
- + notify Observers(): void
- + get Function Name(): String
- + clear(): void

## Value Observer

- \_subject: Value
- + detach (Value): void
- + update(): void

<<enum>>

Type

None,  
String,  
Integer,

## Cell

- \_line: int
- \_column: int
- \_expression: String
- + set Expression (String): void
- + copy(): Cell
- + getLine(): int
- + getColumn(): int
- + paste (Cell): void

## Range

- curr\_line: int
- curr\_col: int
- last\_line: int
- last\_col: int

- + hasNext(): boolean
- + next(): Cell

## Cut Buffer

- buffer: List<Cell>
- length: int
- height: int

- + clear(): void
- + ~~set(String)~~
- + set(Range): void
- + getBuffer(): List<Cell>

xxL

## Spreadsheet

- lines: int
- columns: int
- cells: List<List<Cell>
- users: Collection<String>
- filename: String
- ~~buffer: CutBuffer~~

- + getCell(int, int): Cell
- + getCell(String): Cell
- + getRange(String): Iterator<Cell>
- + getRangeCollection(String): Collection<Cell>

- + copyBuffer(String): void
- + cutBuffer(String): void
- + pasteBuffer(String): void
- + showBuffer(): Collection<Cell>
- + ~~evaluateValue(Value)~~
- + evaluateExpression(Value, String): void
- + insertExpression(String, String): void
- + insertContents(String, String): void
- + searchValue(String): Collection<Cell>
- + searchFunction(String): Collection<Cell>
- + clearCell(String): void
- + addUser(User): boolean
- + removeUser(User): boolean

## Calculator

- spreadsheet: Spreadsheet
- data: DataStore
- ~~user: User~~

- + save(): void
- + saveAs(String): void
- + load(String): void
- + importFile(String): void
- + changeUser(String): void
- + newUser(String): void

xxL

## DataStore

- users: Map<String, User>
- ~~spread: Set<String>~~

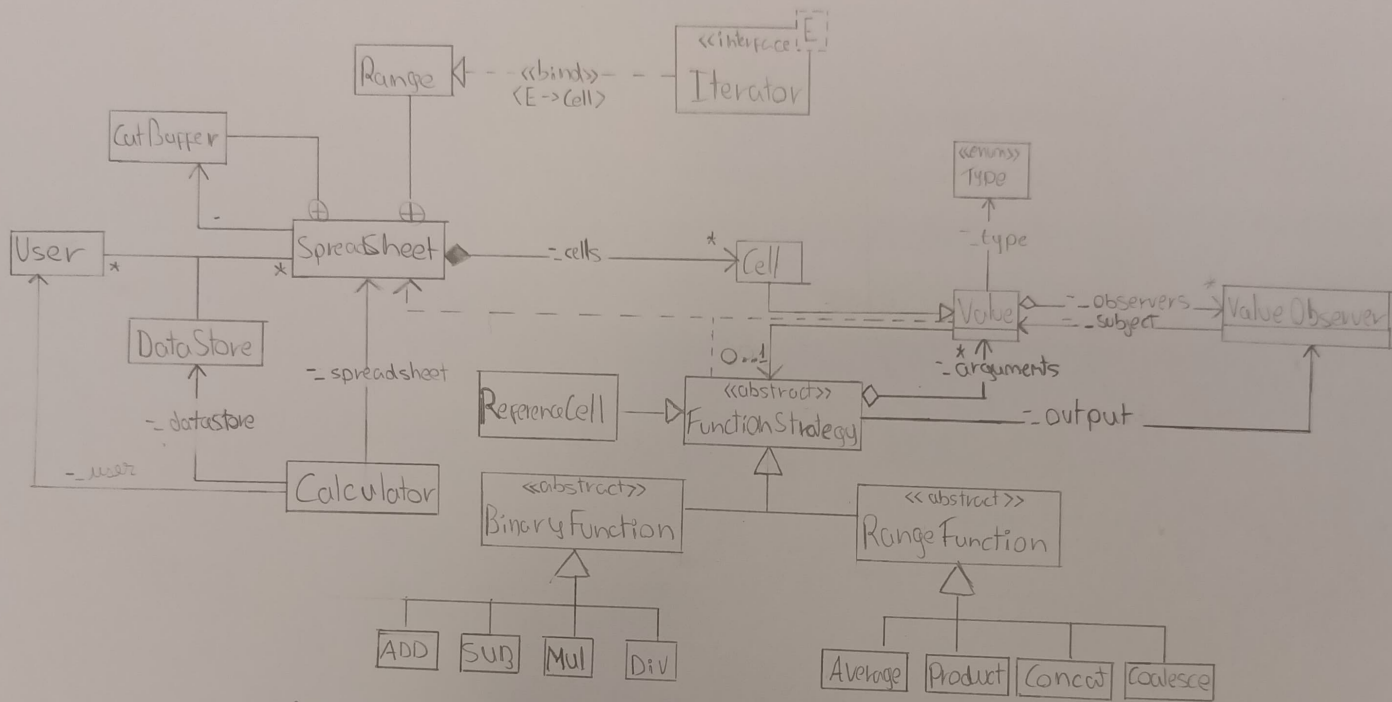
- + add User(User): void
- + add Spread(Spreadsheet, User...): void
- + getUser(String): User
- + remove User

## User

- spreadsheets: Collection<String>
- ~~name: String~~

- + add Spread(String): void
- + rename Spread(String, String): void
- + getSpreads(): Collection<String>
- + remove Spread(String):

xxL, data



Declaro por minha honra que este diagrama foi realizado apenas pelos elementos que constituem o grupo do projeto. Gabriel Ferreira

Declaro por minha honra que este diagrama foi realizado apenas pelos elementos que constituem o grupo do projeto. Francisco Fernandes