HENRIK STAREFORS

SOFTWARE ARCHITECT & BACKEND DEVELOPER

CALLISTA

THE FUTURE OF GENERICS IN GO

HENRIK STAREFORS

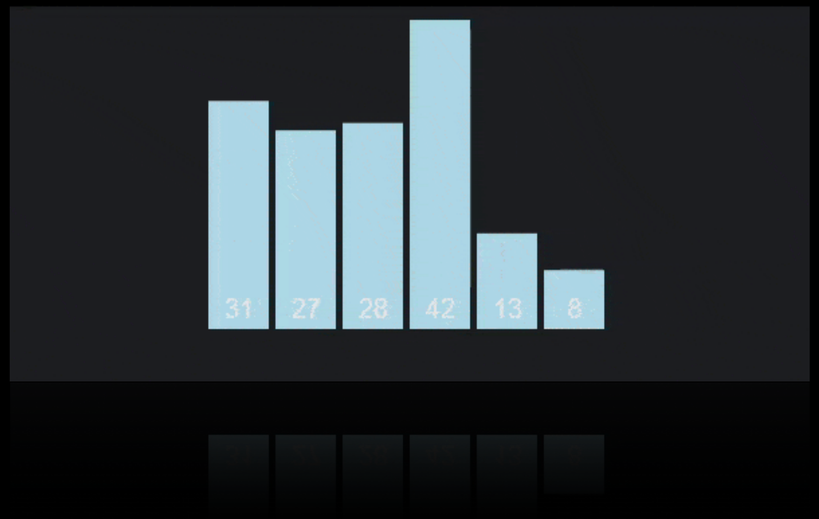
CADEC 2021.01.27 | CALLISTAENTERPRISE.SE

CALLISTA

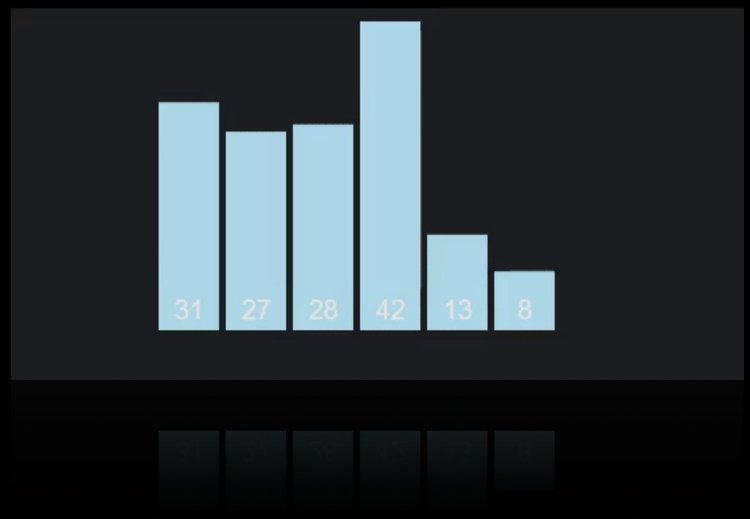
WHY GENERICS?

SORTING

Bubble sort



Quick sort



BUBBLE SORT PYTHON

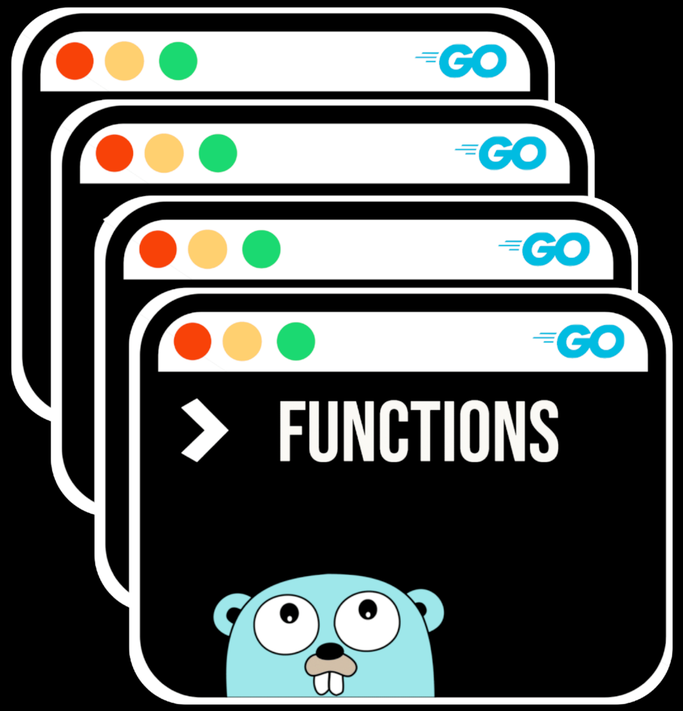
```python
def bubble_sort(data):
    for i in range(0,len(data)-1):
        for j in range(len(data)-1):
            if(data[j]>data[j+1]):
                temp = data[j]
                data[j] = data[j+1]
                data[j+1] = temp
    return data
```

BUBBLE SORT JAVA

```java
static <T extends Comparable<T>> T[] bubbleSort(T[] array){
  for(int i = array.length; i > 1; i--){
    for(int j = 0; j < i - 1; j++){
      if(array[j].compareTo(array[j+1]) > 0){
        T temp = array[i];
        array[i] = array[i+1];
        array[i+1] = temp;
      }
    }
  }
  return array;
}
```

GO ALTERNATIVES

Multiple functions
&
code generation



Interfaces
&
Reflection

BUBBLESORT INT

```
func bubbleSort(input []int) {
  n := len(input)
  swapped := true
  for swapped {
    swapped = false
    for i := 1; i < n; i++ {
      if input[i-1] > input[i] {
        input[i], input[i-1] = input[i-1], input[i]
        swapped = true
      }
    }
  }
}
```

BUBBLESORT STRING

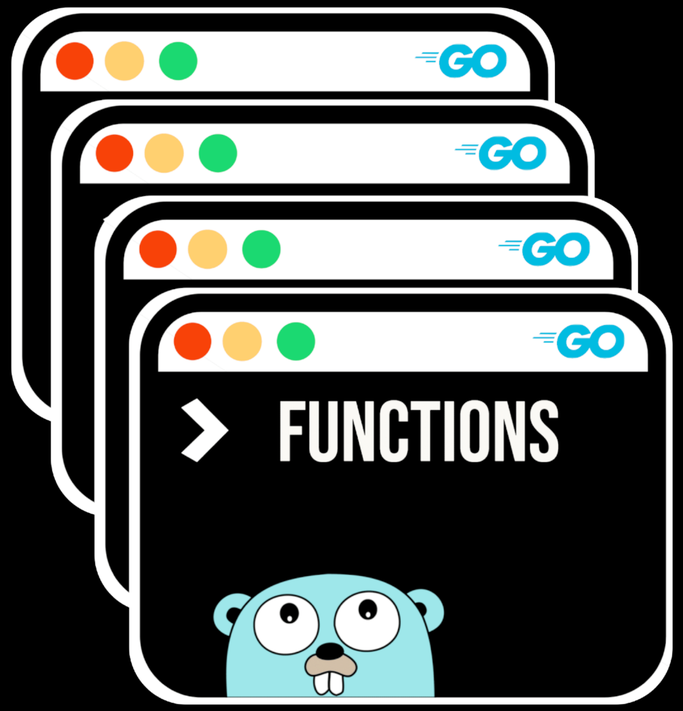
```
func bubbleSort(input []string) {
  n := len(input)
  swapped := true
  for swapped {
    swapped = false
    for i := 1; i < n; i++ {
      if input[i-1] > input[i] {
        input[i], input[i-1] = input[i-1], input[i]
        swapped = true
      }
    }
  }
}
```

BUBBLESORT REFLECTION / INTERFACE

```
func bubbleSort(input []interface{}) {
  data := reflect.ValueOf(input)
  n := reflect.ValueOf(s).Len()

  switch v := input[0].(type) {
  case int:
    // …
  case string:
    // …
  default:
    // …
  }
}
```

GO ALTERNATIVES

Multiple functions
&
code generation



Interfaces
&
Reflection

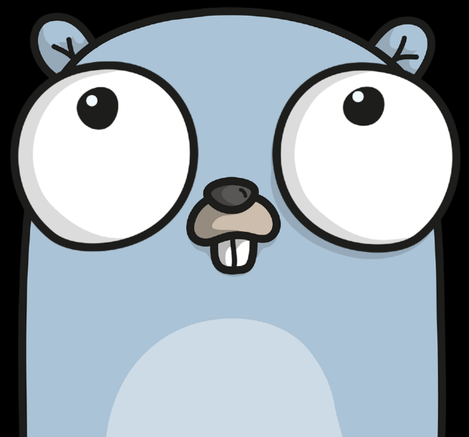
FROM THE GOLANG-NUTS MAILING LIST

Date: Wed, 11 Nov 2009 04:21:45 -0800 PST

IMHO go does not provide two features
a modern programming language needs to provide :

1.Exceptions
2.Templates / Generics

DESIGN PROPOSAL: TYPED PARAMETERS



CALLISTA

PROPOSAL

- Functions can have an additional type parameter list

```
func bubbleSort[T any](input []T) {
  …
}
```

PROPOSAL

- Each Type Parameter has a constrain, just as each regular parameter has a type

```
type numeric interface {
  type int, int8, int16, int32, int64, float32, float64
}


func bubbleSort[T numeric](input []T) {
  …
}
```

PROPOSAL

- Generic functions can only use operations permitted by the type constraints

```
type numeric interface {
  type int, int8, int16, int32, int64, float32, float64
}


func bubbleSort[T numeric](input []T) {
  for _, element := range input {
    element.string() // Invalid
  }
}
```

PROPOSAL

- Type constraints are interface types

```
type Stringer interface {
  String() string
}



func bubbleSort[T Stringer](input []T) {
  for _, element := range input {
    element.string()
  }
}
```

PROPOSAL

- Comparable types in constraints

```
func bubbleSort[T comparable](input []T) {
  for i := 1; i < len(input); i++ {
    If input[i-1] == input[i-1] {
      …
    }
  }
}
```

PROPOSAL

- Comparable types in constraints

```
type ComparableHasher interface {
  comparable
  Hash() uintptr
}



func bubbleSort[T ComparableHasher](input []T) {
 for i := 1; i < len(input); i++ {
    If input[i-1] == input[i-1] {
      …
    }
  }
}
```

PROPOSAL

- Using a generic function or type requires passing type argument

```
type numeric interface {
  type int, int8, int16, int32, int64, float32, float64
}


func bubbleSort[T numeric](input []T) {
  …
}

bubbleSort([]int{1, 2, 3, 4, 5})
bubbleSort([]float64{1.5, 2.5, 3.5, 4.5, 5.5})
```

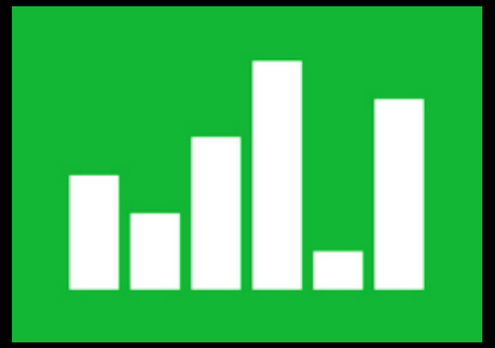
PROPOSAL

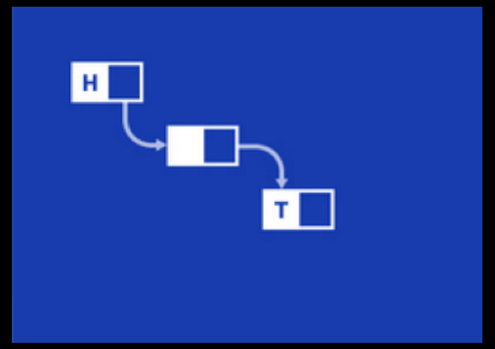
- Types can also have a type parameter list

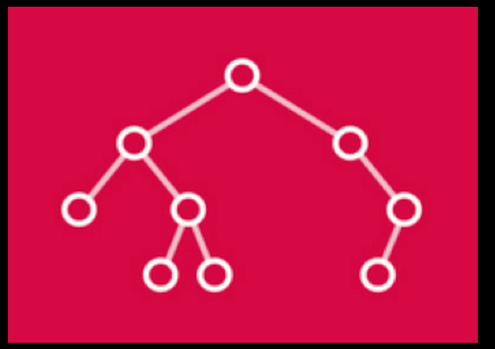
```
type Vector[T any] []T

func (v *Vector[T]) Push(x T) {
  *v = append(*v, x)
}

var a Vector[string]

a.Push("Cadec")

var b Vector[int]

b.Push(2021)
```

DATA STRUCTURES

- Sets
- Containers
- Generic operations on Goroutines
- Multimaps, with multiple key-instances
- Concurrent hash maps



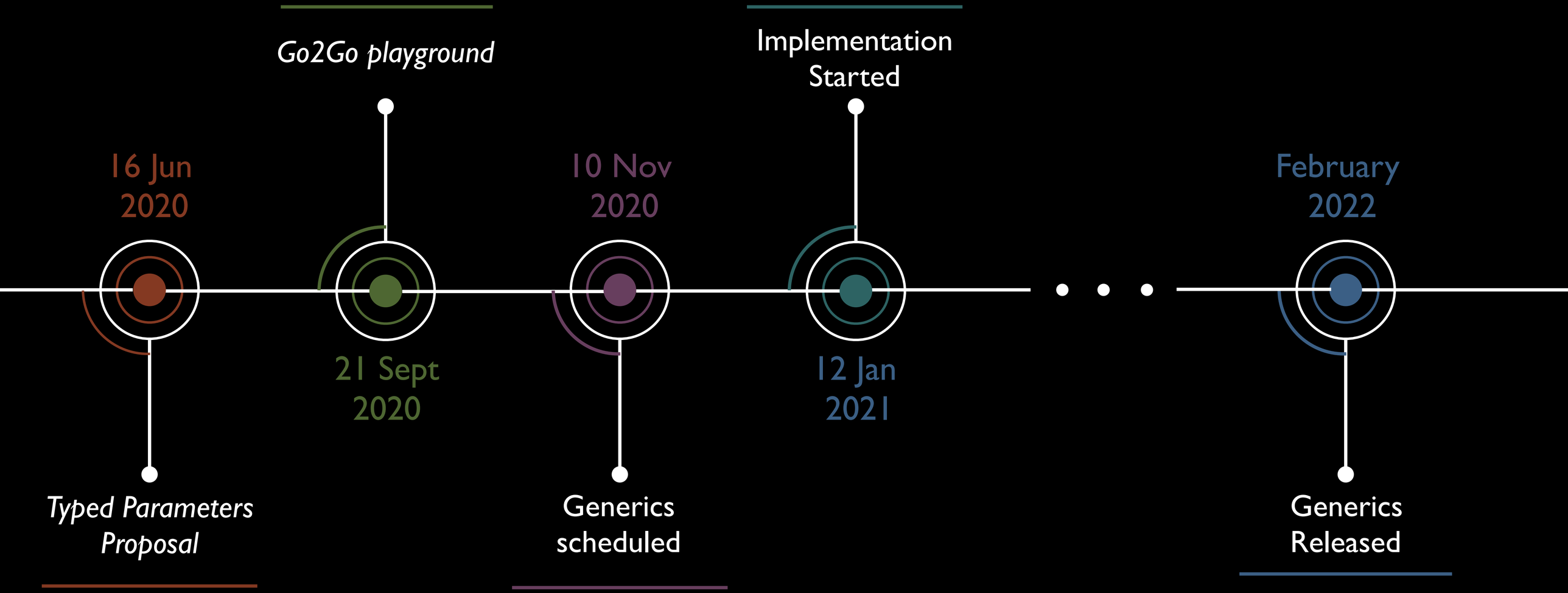




CONCERNS

- Will this make Go more complex? Remove the go idiomatic stance of only having one way to do things.
- Will the barrier of entry be steeper?
- More arguing with the compiler? Slower to read/write more complex code?
- Performance impact?

TIMELINE


16 Jun
2020
Typed Parameters
Proposal
21 Sept
2020
Go2Go playground
10 Nov
2020
Generics
scheduled
12 Jan
2021
Implementation
Started
February
2022
Generics
Released

THANK YOU!

FURTHER READING

- https://blog.golang.org/generics-next-step
- https://go.googlesource.com/proposal/+/refs/heads/master/design/go2draft-type-parameters.md
- https://blog.golang.org/11years
- https://blog.golang.org/generics-proposal
- https://go2goplay.golang.org/