Stint

Jiang Wu, Ningning Xia, Sichang Li, Tingting Ai, Yiming Xu

String and Integer Language

Outline

- Motivation
- Language Features
- Tutorial with Scenario
- Structure Design
- Summary
- Lessons

Background and Motivation

- Strings are mixtures of character blocks and numbers
- Usually we want to manipulate different parts of a string according to what they are
- Existing languages are complicated or heavy-weighted

That's what we focus on!

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Auto-Conversion:

```
string str1 = 12;          -> "12"
string str2 = true;        -> "true"
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Int-Extraction:

"age 22, grade 98, rank 3"

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Int-Extraction:

"age 23, grade 98, rank 3"

```
str .<| 0 |> = str .<| 0 |> + 1;
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Insert & Delete:

```
"this is "

str = str + "Stint";
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Insert & Delete:


```
str = str + "Stint";
str = str + "not " @ 8;
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Insert & Delete:

```
"this is not Stint"

str = str + "Stint";
str = str + "not " @ 8;
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Insert & Delete:

"this is not Stint"

```
str = str - "is";        "this is not Stint"
str = str - "is" @ 4;    "this is not Stint"
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

Extraction:

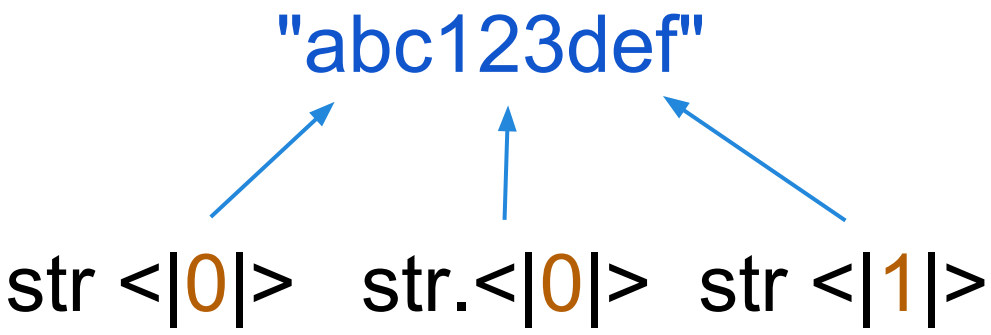
"abc123def"

str[0] -> "a"
str[0, 3] -> "abc"

Language Features

* Flexible Type Conversion
* More Convenient String Operations

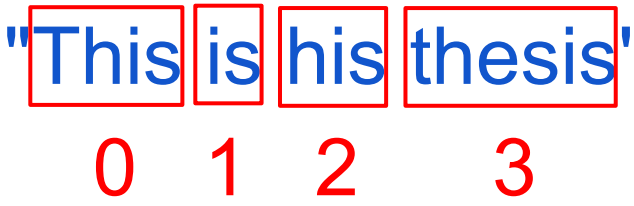
Extraction:



Language Features

* Flexible Type Conversion
* More Convenient String Operations

More Extraction:

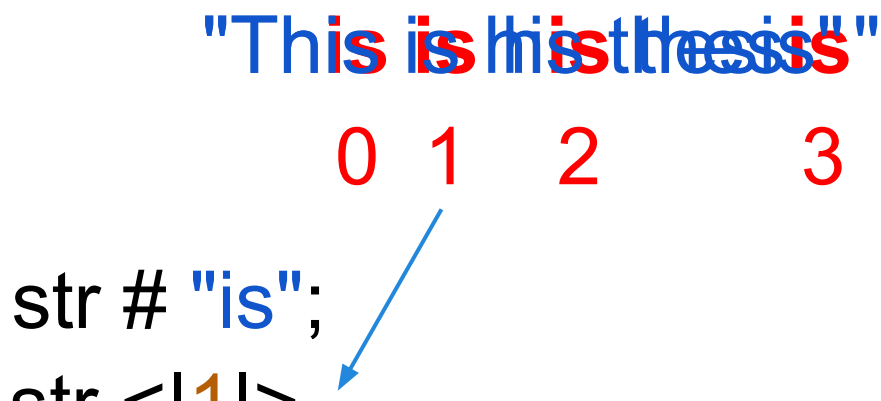



```
str | " ";
str <|2|>     -> "his"
```

Language Features

* Flexible Type Conversion
* More Convenient String Operations

More Extraction:



Language Features

* Flexible Type Conversion
* More Convenient String Operations

Assignment:

"This isn't this" "It's this"

0 1 2 3

```
str # "is";
str <|1|> = "isn't"
```

Tutorial with Scenario

Update inventory record of a store:

sn1_table.txt :

```
<name><price><sales><inventory>
MacBookPro $1200 248 203
MacBookAir $900 381 246
iMac $1600 64 98
.....
```

sn1_input.txt :

```
<sales in this month>
194
246
63
.....
```

sn1_output.txt :

```
<name><mark><price><sales><inventory><income>
MacBookPro(BEST) $1200 442 9 530400
MacBookAir(BEST) $900 627 0 564300
iMac $1600 127 35 203200
.....
```

Tutorial with Scenario

```
void main () {
        string input_file = "sn1_table.txt";
        ..... /* save all file names as variables */
        string in_buff = "";
        string data_buff = "";
        int sales;

        /* open input and output files */
        open input_file; .....

        /* read line by line */
        while (input_file >> in_buff) {
                data_file >> data_buff;
                sales = data_buff.<|0|>;

        /* modify sales and inventory */
        in_buff.<|1|> = in_buff.<|1|> + sales;
        in_buff.<|2|> = in_buff.<|2|> - sales;

        /* calculate total sale income */
        in_buff = in_buff + " " + (in_buff.<|0|> * in_buff.<|1|>);

        /* mark tag if necessary */
        in_buff | " ";
        if (in_buff.<|3|> > 300000) {
                in_buff<|0|> = in_buff<|0|> + "(BEST)"; }
                if (in_buff.<|3|>  < 100000) {
                        in_buff<|0|> = in_buff<|0|> + "(OFFER)";
                        in_buff.<|0|> = in_buff.<|0|> / 2;
                }
                output_file << in_buff + "\n";
        }

        close input_file; .....   /* close all opened files */

        return;  /* must have a return statement */
        }
```

Tutorial with Scenario

```
void main () {
    string input_file = "sn1_table.txt";
    ..... /* save all file names as variables */
    string in_buff = "";
    string data_buff = "";
    int sales;

    /* open input and output files */
    open input_file; .....
```

```
<name><price><sales><inventory>
MacBookPro $1200 248 203
MacBookAir $900 381 246
iMac $1600 64 98
.....
```

```
<sales in this month>
194
246
63
.....
```

Tutorial with Scenario

```
void main () {
    string input_file = "sn1_table.txt";
    ..... /* save all file names as variables */
    string in_buff = "";
    string data_buff = "";
    int sales;

/* read line by line */
    while (input_file >> in_buff) {
    data_file >> data_buff;
    sales = data_buff.<|0|>;

    /* modify sales and inventory */
    in_buff.<|1|> = in_buff.<|1|> + sales;
    in_buff.<|2|> = in_buff.<|2|> - sales;
```

```
/* calculate total sale income */
in_buff = in_buff + " " + (in_buff.<|0|> * in_buff.<|1|>);

/* mark tag if necessary */
in_buff | " ";
if (in_buff.<|3|> > 300000) {
    in_buff<|0|> = in_buff<|0|> + "(BEST)"; }
```

data_buff:
194

```
output_file << in_buff + "\n";
```

in_buff:
MacBookPro $1200 248 203

MacBookPro $1200 442 9

Tutorial with Scenario

```
void main () {
    string input_file = "sn1_table.txt";
    ..... /* save all file names as variables */
    string in_buff = "";
    string data_buff = "";
    int sales;
```

in_buff:
MacBookPro $1200 248 203

⇩

MacBookPro $1200 442 9 530400

⇩

MacBookPro(BEST) $1200 442 9 530400

```
/* calculate total sale income */
in_buff = in_buff + " " + (in_buff.<|0|> * in_buff.
<|1|>);

/* mark tag if necessary */
in_buff | " ";
if (in_buff.<|3|> > 300000) {
    in_buff<|0|> = in_buff<|0|> + "(BEST)"; }
    if (in_buff.<|3|>  < 100000) {
        in_buff<|0|> = in_buff<|0|> + "(OFFER)";
        in_buff.<|0|> = in_buff.<|0|> / 2;
    }
    output_file << in_buff + "\n";
}
```

Tutorial with Scenario

```
void main () {
    s
    .
    s
    s
    i
    /* open input and outpu
    open input_file; ......

    /                    ine */
    while (input_file >> in_b
    data_file >> data_buff;
    sales = data_buff.<|0|>;

    /* modify sales and inventory */
    in_buff.<|1|> = in_buff.<|1|> + sales;
    in_buff.<|2|> = in_buff.<|2|> - sales;
```

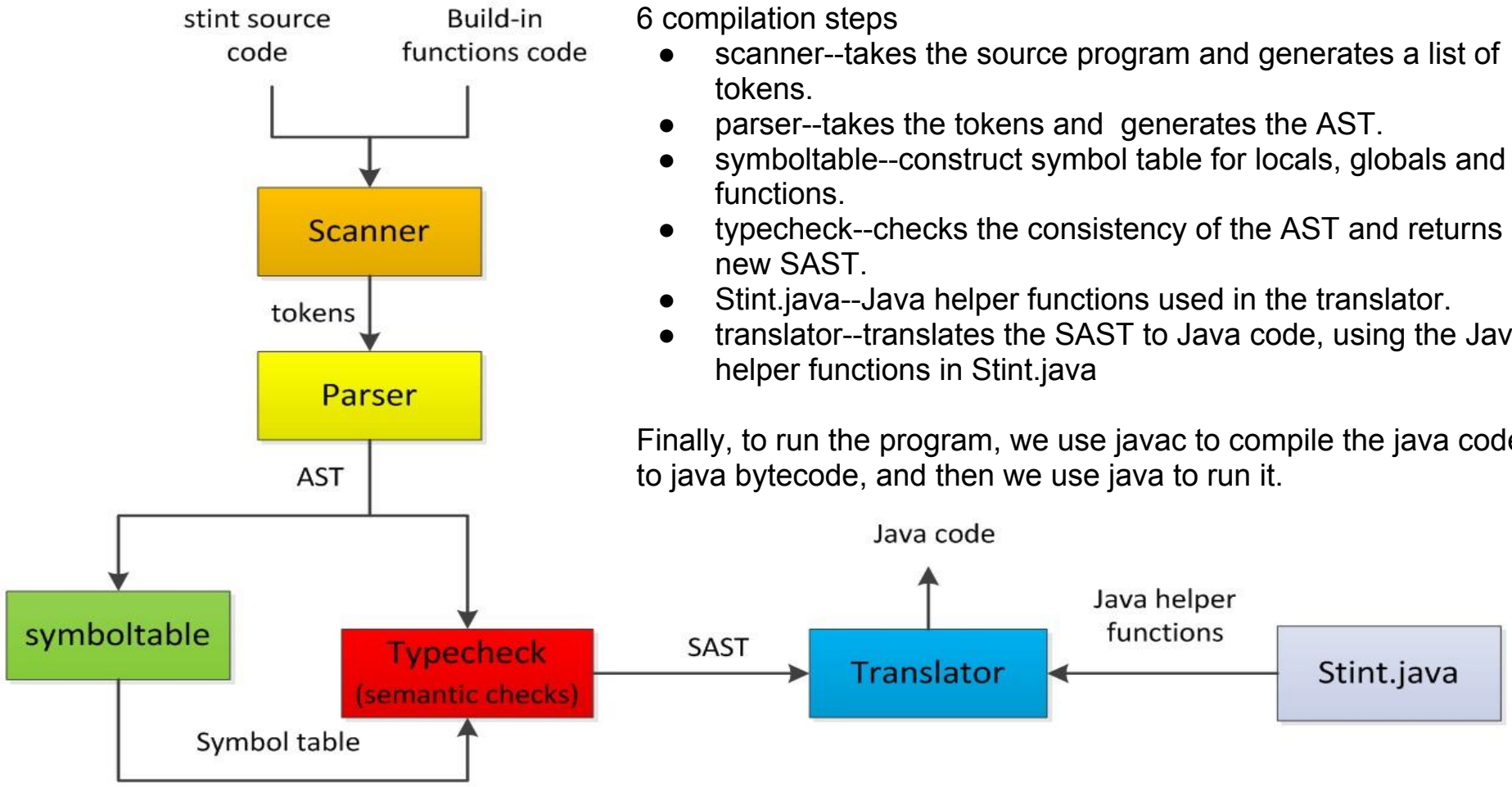
```
<name><price><sales><inventory>
MacBookPro $1200 248 203
MacBookAir $900 381 246
iMac $1600 64 98
......
```

```
<name><mark><price><sales><inventory><income>
MacBookPro(BEST) $1200 442 9 530400
MacBookAir(BEST) $900 627 0 564300
iMac $1600 127 35 203200
......
```

```
    * calculate total sale income */
    n_buff = in_buff + " " + (in_buff.<|0|> * in_buff.<|1|>);

    * mark tag if necessary */
    n_buff | " ";
    if (in_buff.<|3|> > 300000) {
        in_buff<|0|> = in_buff<|0|> + "(BEST)"; }
                                                    R)";

    close input_file; .....   /* close all opened files */

    return;  /* must have a return statement */
    }
```

Structure Design

Stint compiler structure



6 compilation steps

- scanner--takes the source program and generates a list of tokens.
- parser--takes the tokens and generates the AST.
- symboltable--construct symbol table for locals, globals and functions.
- typecheck--checks the consistency of the AST and returns a new SAST.
- Stint.java--Java helper functions used in the translator.
- translator--translates the SAST to Java code, using the Java helper functions in Stint.java

Finally, to run the program, we use javac to compile the java code to java bytecode, and then we use java to run it.

Summary

- **Goals achieved**

  - Simplicity: Use operators to provide basic string operations
  - Convenience: Implementation of I/O functions and type conversion
  - A strongly typed language

- **Challenges**

  - Unfamiliarity with formal grammar
  - Translation from Ocaml to Java
  - Data structure design

Lessons & Tools

- **Lessons**

  - Planning and scheduling
  - Thinking and working in functional language
  - Team cooperation -share ideas -communicate

- **Tools**

  - Ocaml
  - Eclipse
  - Git - Version Control
  - Google Doc - Documentation

Q&A

Thanks