Session 04:

String, Regex

NỘI DUNG

1. Giới thiệu chuỗi, khai báo chuỗi
2. Các phương thức làm việc với chuỗi
3. Giới thiệu lớp StringBuilder vs StringBuffer
4. Giới thiệu về biểu thức chính quy

1. Giới thiệu chuỗi, khai báo chuỗi

Chuỗi là gì?

- Là tập hợp các ký tự
- Giá trị của chuỗi là tập hợp một hoặc nhiều ký tự trong dấu ngoặc kép “ ”

Trong Java có 3 lớp xử lý chuỗi trong Java:

String

StringBuilder

StringBuffer

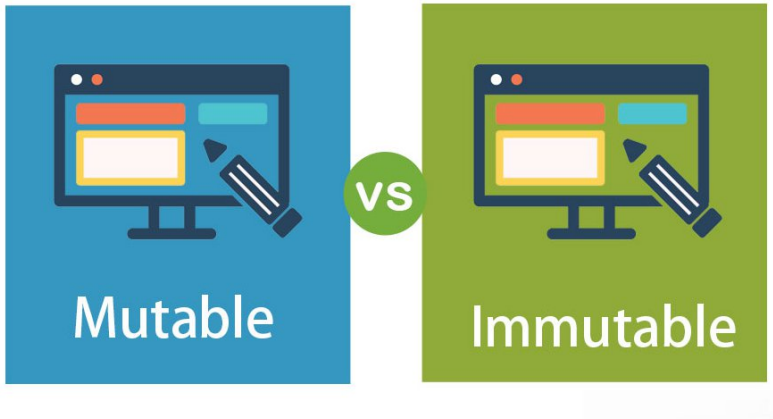
1. Giới thiệu chuỗi, khai báo chuỗi

immutable:

- *Mutable Object*: trạng thái của đối tượng có thể thay đổi được sau khi khởi tạo đối tượng thành công

mutable:

- *Immutable Object*: trạng thái của đối tượng không thể thay đổi được sau khi khởi tạo đối tượng thành công



1. Giới thiệu chuỗi, khai báo chuỗi

String là một lớp đặc biệt vừa có tính nguyên thủy (primitive) vừa có tính đối tượng (reference)

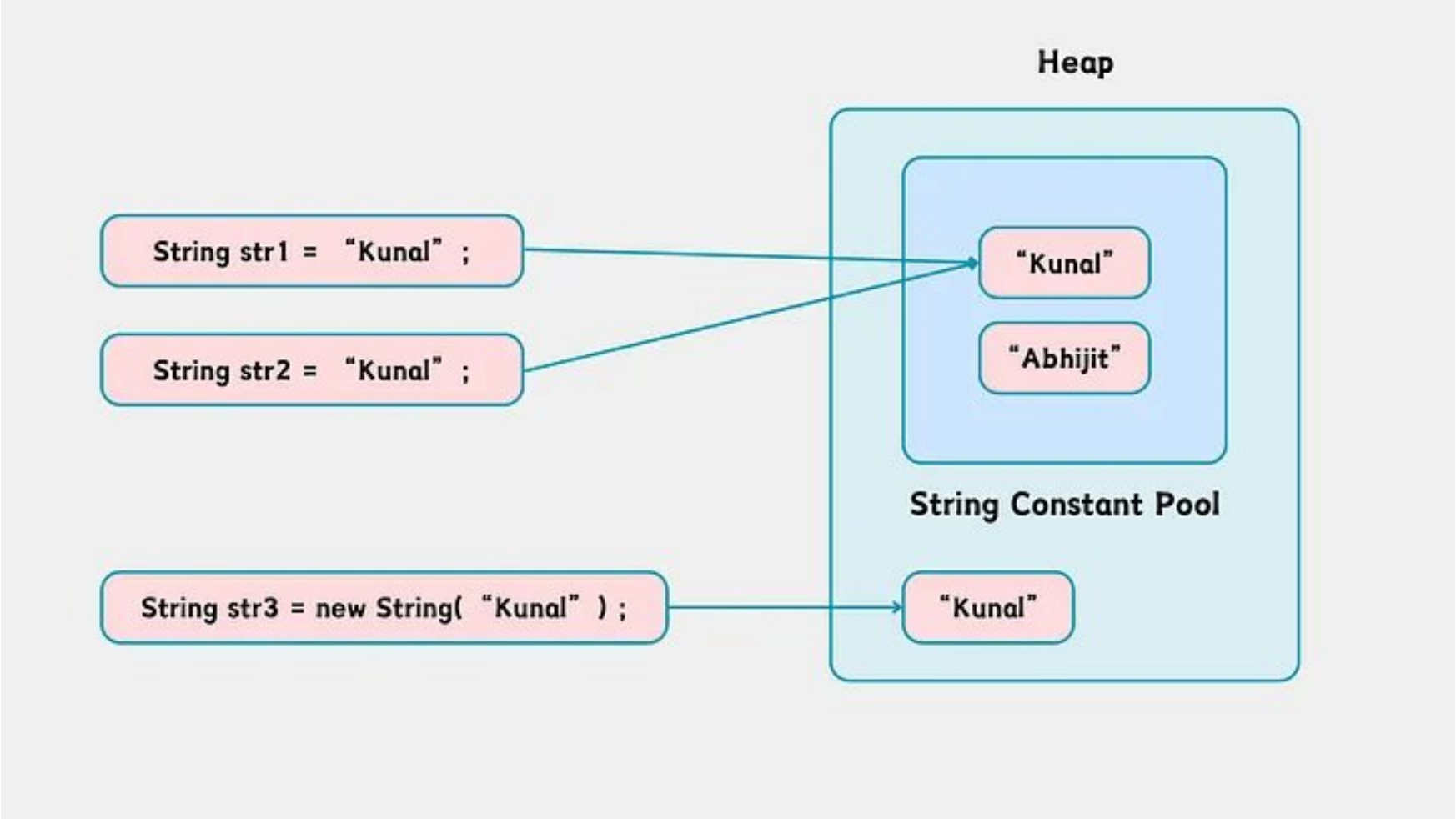
(nguyên thuỷ) primitive

```
String str = "Primitive";
```

(đối tượng) reference

```
String str = new String("References");
```

1. Giới thiệu chuỗi, khai báo chuỗi



2. Các phương thức làm việc với chuỗi

Các phương thức cơ bản:

Phương thức	Mô tả
int length()	Trả về độ dài (số ký tự) của chuỗi
Char charAt(int index)	Trả về một ký tự tại chỉ số index
String concat(String str)	Nối chuỗi vào cuối chuỗi
String toUpperCase()	Viết hoa tất cả ký tự của chuỗi

Phương thức	Mô tả
String toLowerCase()	Viết thường tất cả ký tự chuỗi
String trim()	Trả về chuỗi mới sau khi loại bỏ các ký tự trắng 2 đầu
boolean equals (String str)	So sánh bằng 2 đối tượng chuỗi
boolean equalsIgnoreCase (String str)	So sánh 2 đối tượng chuỗi không phân biệt hoa thường

2. Các phương thức làm việc với chuỗi

Các phương thức làm việc với chuỗi:

Phương thức	Mô tả
int compareTo (String str)	So sánh 2 chuỗi
int compareTolgnoreCase (String str)	So sánh 2 chuỗi không phân biệt hoa thường
boolean contains (String str)	Kiểm tra trong chuỗi có chứa str
int indexOf(String str)	Trả về chỉ số của str trong chuỗi
int lastIndexOf(String str)	Trả về chỉ số chuỗi cùng của str trong chuỗi
String replace (char oldChar, char newChar)	Thay thế tất cả oldChar thành newChar
String replaceAll(String regex, String replacement)	Thay thế tất cả chuỗi con khớp với regex thành replacement

2. Các phương thức làm việc với chuỗi

Các phương thức làm việc với chuỗi:

Phương thức	Mô tả
boolean startsWith (String prefix)	Kiểm tra chuỗi có bắt đầu là prefix
boolean endsWith (String suffix)	Kiểm tra chuỗi có kết thúc là suffix
String[] split(String regex)	Tách chuỗi thành chuỗi con
String substring (int beginIndex)	Cắt chuỗi từ chỉ số beginIndex cho đến cuối
String substring(int beginIndex, int endIndex)	Cắt chuỗi từ chỉ số beginIndex đến chỉ số endIndex
char[] toCharArray()	Chuyển chuỗi thành mảng ký tự

2. Các phương thức làm việc với chuỗi

Cho 3 chuỗi sau:

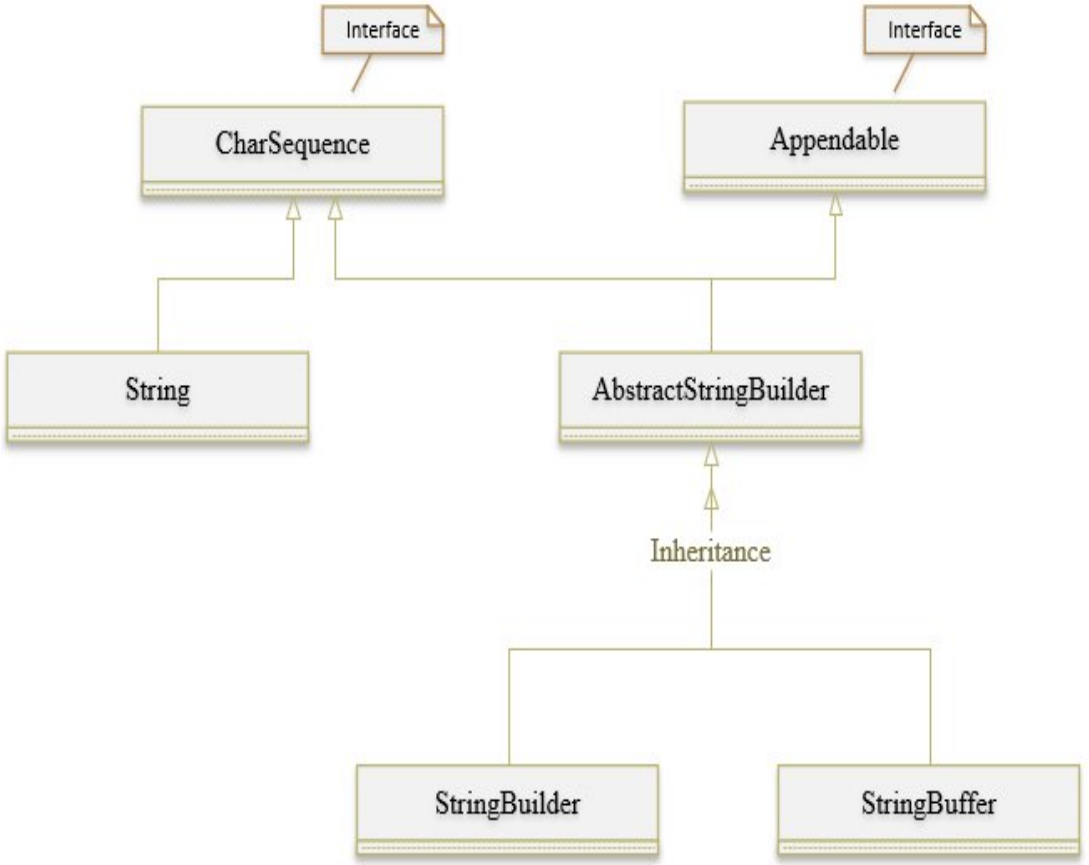
```
String s1 = "Welcome to JAVA";
String s2 = "Programming is fun";
String s3 = new String("Welcome to JAVA");
```

Kết quả của các biểu thức sau:

- s1 == s2
- s1 == s3
- s1.equals(s2)
- s1.equals(s3)
- s1.compareTo(s2)
- s1.compareTo(s3)
- s1.charAt(1)
- s1.indexOf('j')
- s2.indexOf('f')
- s1.lastIndexOf('o')
- s1+s2
- s1.concat(s3)
- s1.replace("o", "a")
- s1.replaceAll("o", "a")
- s1.toUpperCase()
- s1.toLowerCase()
- s1.substring(5)
- s1.substring(5,8)
- s1.startsWith("welcome")
- s1.endsWith("JAVA")
- s1.contains(s2)
- s1.length()
- s1.split()
- s1.split(" ")

3. Giới thiệu lớp StringBuilder vs StringBuffer

- **StringBuilder**
  - **Không đồng bộ (not synchronized)**
  - **Tốc độ nhanh**
  - Dùng trong **môi trường đơn luồng (single-thread)**
- **StringBuffer**
  - **Có đồng bộ (synchronized)**
  - **An toàn trong đa luồng**
  - Tốc độ chậm hơn **StringBuilder**



3. Giới thiệu lớp StringBuilder vs StringBuffer

Khởi tạo đối tượng StringBuilder vs StringBuffer

```
StringBuilder stringBuilder = new StringBuilder();
```

```
StringBuffer stringBuffer = new StringBuffer();
```

3. Giới thiệu lớp StringBuilder vs StringBuffer

Rikkei Academy
where the dream begins

Các phương thức làm việc với StringBuilder vs StringBuffer

Phương thức	Mô tả	Ví dụ ngắn
append()	Nối chuỗi / Dữ liệu	sb.append(“Java”)
insert()	Chèn chuỗi tại một vị trí	sb.insert(4, “ Pro”)
delete()	Xóa một đoạn chuỗi	sb.delete(3, 4)
replace()	Thay thế chuỗi	sb.replace(0, 4, “Python”)
reverse()	Đảo ngược chuỗi	sb.reverse()

3. Giới thiệu lớp StringBuilder vs StringBuffer

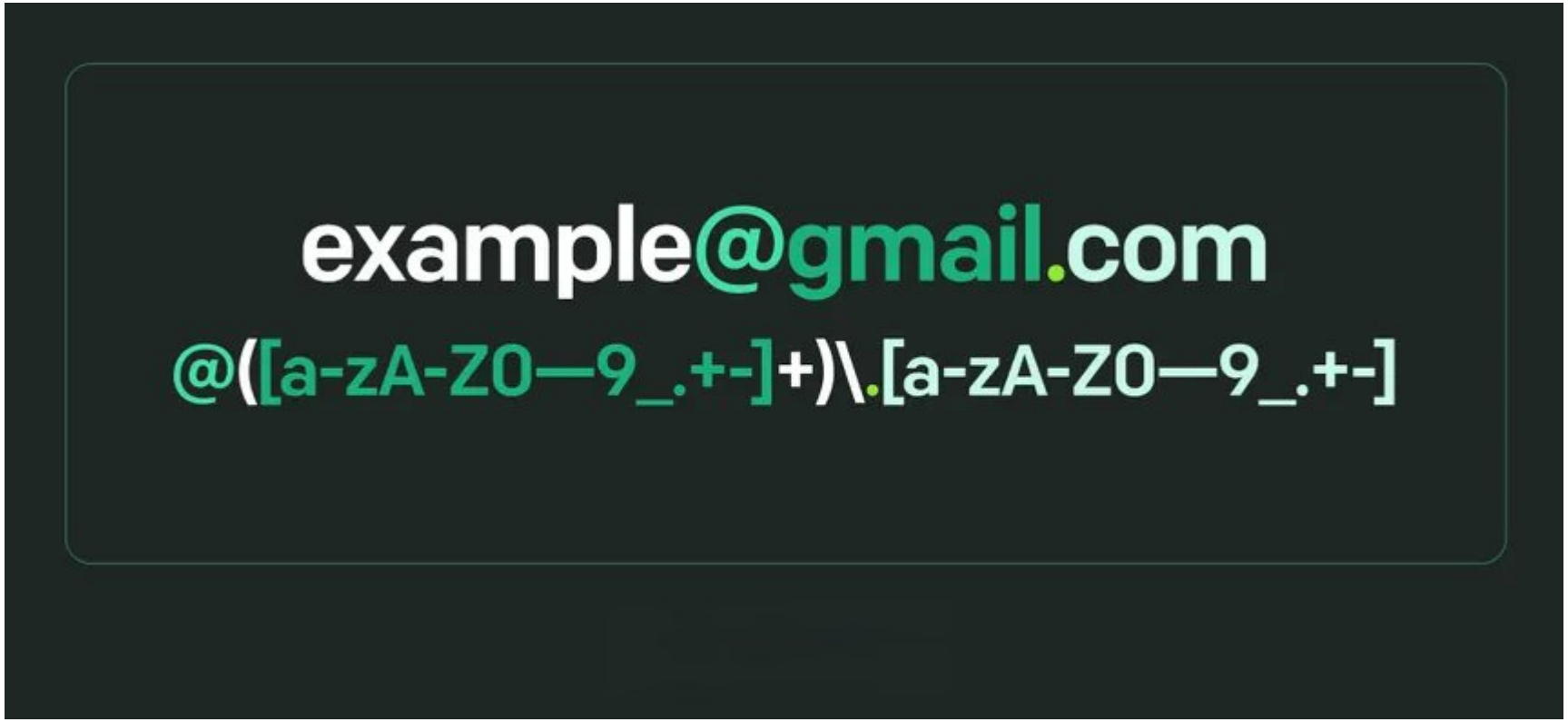
Rikkei Academy
where the dream begins

Các phương thức làm việc với StringBuilder vs StringBuffer

Phương thức	Mô tả	Ví dụ ngắn
length()	Lấy ra độ dài	sb.length()
charAt()	Lấy ký tự tại vị trí trong chuỗi	sb.charAt(3)
toString()	Biến thành chuỗi (String)	sb.toString()
setLength()	Xóa / Thay đổi độ dài	sb.setLength(0)

4. Giới thiệu về biểu thức chính quy

Regular Expression – Biểu thức chính quy viết tắt là **Regex** là một chuỗi mẫu được sử dụng để quy định dạng thức của các chuỗi



4. Giới thiệu về biểu thức chính quy

Phương thức ***matches***() thuộc lớp Pattern trong Java xác định có hay không chuỗi này so khớp với regular expression đã cho

Cú pháp:

```
boolean isMatch = Partern.matches(String regex, CharSequence input)
```

Trong đó:

- **regex**: Regular expression là biểu thức chính quy dùng để so khớp
- **input**: chuỗi đầu vào cần kiểm tra

4. Giới thiệu về biểu thức chính quy

Các cách sử dụng phương thức matches()

```
Pattern pattern = Pattern.compile("^[a-zA-Z0-9]$");
Matcher matcher = pattern.matcher(input);
boolean isMatch = matcher.matches();
```

```
boolean isMatch = Pattern.compile("^[a-zA-Z0-9]$")
                         .matcher(input)
                         .matches();
```

```
boolean isMatch = Pattern.matches("^[a-zA-Z0-9]$",input);
```

4. Giới thiệu về biểu thức chính quy

Quy tắc viết biểu thức chính quy:

Regex	Matches	Example
x	Chỉ định rõ là một ký tự x	Java matches “Java”
.	Đại diện 1 ký tự bất kỳ	Java matches “J..a”
(ab | cd)	Là ab hoặc cd	Java matches “Ja(va|em)”
[abc]	Ký tự a hoặc b hoặc c	Java matches “Ja[uvwx]a”
[^abc]	Bất cứ ký tự nào khác a và b và c	Java matches “Ja[^abc]a”
[a-z]	Các ký tự từ a đến z	Java matches “[A-M]av[a-d]”
[^a-z]	Các ký tự không nằm trong khoảng a-z	Java matches “Jav[^b-d]”
[a-e[m-p]]	Ký tự nằm trong khoảng a-e hoặc m-p	Java matches “[A-G[I-M]]ava”
[a-e&&[c-p]]	Nằm trong khoảng giao của a-e và c-p	Java matches “[A-P&&[I-M]]ava”

4. Giới thiệu về biểu thức chính quy

Quy tắc viết biểu thức chính quy:

Regex	Matches	Example
\d	Là một số tương đương [0-9]	Java2 matches “Java[\\d]”
\D	Không phải là 1 số	Java matches “[\\D]ava”
\w	Là một ký tự	Java1 matches “[\\w]ava[\\w]”
\W	Không phải là một ký tự	$Java matches “[\\W]Java”
\s	Là một khoảng trắng	“Java 2” matches “Java\\s2”
\S	Không phải là một khoảng trắng	Java matches “[\\S]Java”

4. Giới thiệu về biểu thức chính quy

Quy tắc viết biểu thức chính quy:

Regex	Matches	Example
p*	Không, một hoặc nhiều ký tự p	aaaabb matches “a*bb”
p+	Một hoặc nhiều ký tự p	ab matches “a+b”
p?	Không hoặc 1 ký tự p	Java matches “J?Java”
p{n}	Có n ký tự p	Java matches “Ja{1}.*”
p{n,}	Có ít nhất n ký tự p	aaa matches “a{2,}”
p{n,m}	Có từ n-m ký tự p	aaa matches “a{2,9}”

TỔNG KẾT

- Nắm kiến thức về chuỗi và cú pháp khai báo chuỗi
- Hiểu và nắm được các phương thức làm việc với chuỗi
- Hiểu và nắm được về đối tượng StringBuilder
- Giới thiệu về biểu thức chính quy

RIKEI Academy®
where the dream begins

KẾT THÚC

HỌC VIỆN ĐÀO TẠO LẬP TRÌNH CHẤT LƯỢNG NHẬT BẢN