web APIs 第六天

正则表达式、阶段案例

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

学习目标

Learning Objectives

1. 能够利用正则表达式校验输入信息的合法性
2. 具备利用正则表达式验证小兔鲜注册页面表单的能力

目录

Contents

正则表达式

- 基本使用
- 元字符
- 替换和修饰符

- 目标：学习正则表达式概念及语法，编写简单的正则表达式实现字符的查找或检测。

1.1.1 什么是正则表达式

- 正则表达式（Regular Expression）是一种字符串匹配的模式（规则）
- **使用场景：**
  - 例如验证表单：手机号表单要求用户只能输入11位的数字（匹配）
  - 过滤掉页面内容中的一些敏感词(替换)，或从字符串中获取我们想要的特定部分(提取)等

1.1.2 正则基本使用

- 正则基本使用分为两步：

定义规则

- const reg = /表达式/
- 其中 / / 是正则表达式字面量
- 正则表达式也是对象

```
const reg = /前端/
```

使用正则

- **test()** 方法 用来查看正则表达式与指定的字符串是否匹配
- 如果正则表达式与指定的字符串匹配，返回true，否则false

```
// 要检测的字符串
const str = 'IT培训,前端开发培训,IT培训课程,web前端培训,Java培训,人工智能培训'
// 1.定义正则表达式，检测规则
const reg = /前端/
// 2.检测方法
console.log(reg.test(str)) // true
```

总结

1.正则表达式是什么？

- 是一种字符串匹配的模式（规则）

2.正则表达式有什么作用？

- 表单验证（**匹配**）
- 过滤敏感词（**替换**）
- 字符串中提取我们想要的部分（**提取**）

3.正则表达式使用分为几步？

- 定义规则
- 使用正则 test() 方法

```
// 正则表达式的基本使用
const str = 'web前端开发'
// 1. 定义规则
const reg = /web/

// 2. 使用正则 test()
console.log(reg.test(str))  // true  如果符合规则匹配上则返回true
console.log(reg.test('java开发'))  // false  如果不符合规则匹配上则返回
```

正则表达式

- 基本使用
- 元字符
- 替换和修饰符

- 目标：学习正则表达式概念及语法，编写简单的正则表达式实现字符的查找或检测。

1.2 元字符

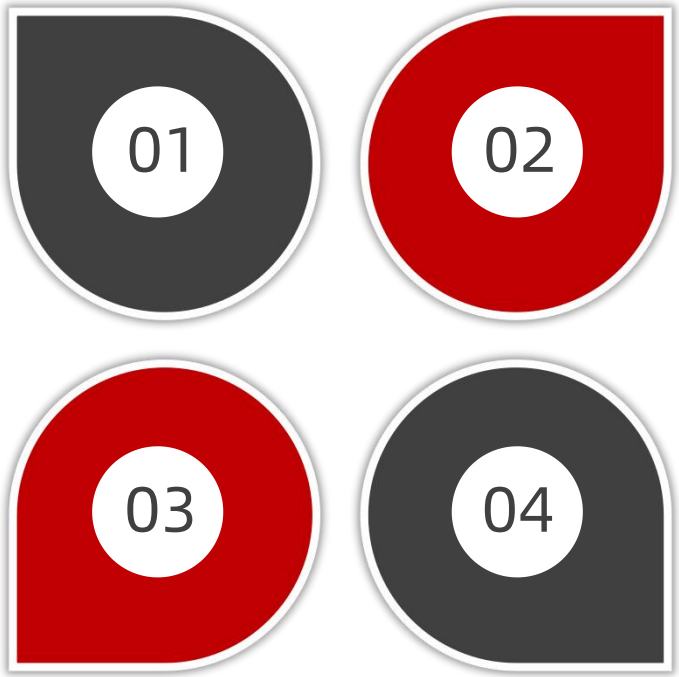
- **普通字符：**
  - 大多数的字符仅能够描述它们本身，这些字符称作普通字符，例如所有的字母和数字。
  - 普通字符只能够匹配字符串中与它们相同的字符。
  - 比如，规定用户只能输入英文26个英文字母，普通字符的话 /[abcdefghijklmnopqrstuvwxyz]/
- **元字符(特殊字符)**
  - 是一些具有特殊含义的字符，可以极大提高了灵活性和强大的匹配功能。
  - 比如，规定用户只能输入英文26个英文字母，换成元字符写法： /[a-z]/

1.2 元字符

- 为了方便记忆和学习，我们对众多的元字符进行了分类：

边界符

定义位置规则，必须用什么开头，用什么结尾



量词

定义重复次数规则

范围

表示字符的范围

字符类

区分各种字符，例如区分字母和数字

1.2 元字符

1. 边界符

- 正则表达式中的边界符（位置符）用来提示字符所处的位置，主要有两个字符

边界符	说明
^	表示匹配行首的文本(以谁开始)
$	表示匹配行尾的文本(以谁结束)

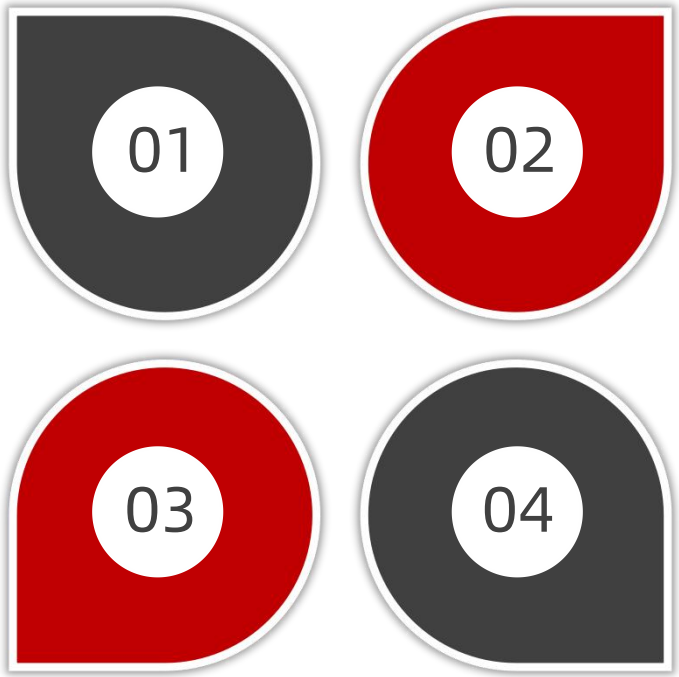
如果 ^ 和 $ 在一起，表示必须是精确匹配

1.2 元字符

- 为了方便记忆和学习，我们对众多的元字符进行了分类：

边界符

定义位置规则，必须用什么开头，用什么结尾



量词

定义重复次数规则

范围

表示字符的范围

字符类

区分各种字符，例如区分字母和数字

1.2 元字符

2. 量词

- 量词用来设定某个模式重复次数

量词	说明
*	重复零次或更多次
+	重复一次或更多次
?	重复零次或一次
{n}	重复n次
{n,}	重复n次或更多次
{n,m}	重复n到m次

注意： 逗号左右两侧千万不要出现空格

1.2 元字符

2. 量词

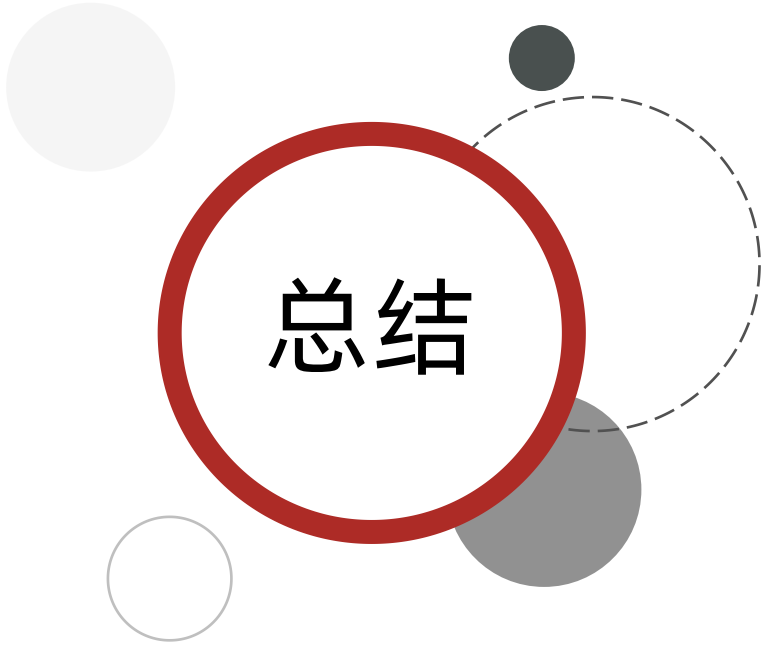
- 量词用来 设定某个模式出现的次数

```
// * 表示重复0次或者更多次
console.log(/^哈*$/.test(''))  // true
console.log(/^哈*$/.test('哈'))  // true
console.log(/^哈*$/.test('哈哈哈'))  // true
// + 表示重复1次或者更多次
console.log(/^哈+$/.test(''))  // false
console.log(/^哈+$/.test('哈'))  // true
console.log(/^哈+$/.test('哈哈哈'))  // true
// ?  表示重复0次或者1次    0 || 1
console.log('------')
console.log(/^哈?$/.test(''))  // true
console.log(/^哈?$/.test('哈'))  // true
console.log(/^哈?$/.test('哈哈哈'))  // fasle
```

```
console.log(/^哈{2}$/.test(''))  // false
console.log(/^哈{2}$/.test('哈'))  // false
console.log(/^哈{2}$/.test('哈哈'))  // true
console.log(/^哈{2}$/.test('哈哈哈'))  // false
```

```
//  {n,}   是 >= n 的意思
console.log(/^哈{2,}$/.test(''))  // false
console.log(/^哈{2,}$/.test('哈'))  // false
console.log(/^哈{2,}$/.test('哈哈'))  // true
console.log(/^哈{2,}$/.test('哈哈哈'))  // true
```

```
//  {n,m}   是  次数  >= n   <= m  的意思  注意，逗号左右千万不要有空格
console.log(/^哈{2,4}$/.test(''))  // false
console.log(/^哈{2,4}$/.test('哈'))  // false
console.log(/^哈{2,4}$/.test('哈哈'))  // true
console.log(/^哈{2,4}$/.test('哈哈哈'))  // true
console.log(/^哈{2,4}$/.test('哈哈哈哈'))  // true
console.log(/^哈{2,4}$/.test('哈哈哈哈哈'))  // false
```



1. 字符串 'p' 想要重复 3次以上怎么写？

- /^p{3,}$/

2. 字符串 'p' 想要重复6~18次之间怎么写？

- /^p{6,18}$/

3. 字符串 'p' 想要重复0次以上怎么写？

- /^p*$/ 或者 /^p{0,}$/

4. 字符串 'p' 想要至少出现一次怎么写？

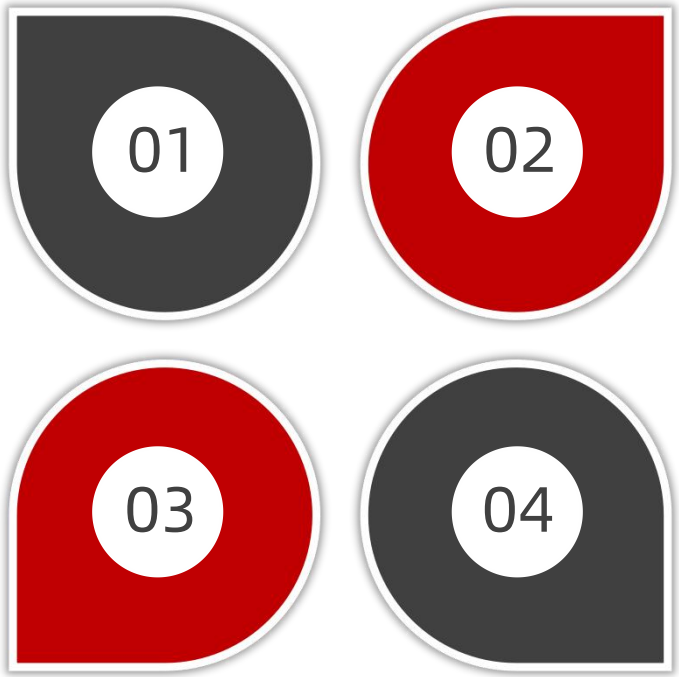
- /^p+$/或者 /^p{1,}$/

1.2 元字符

- 为了方便记忆和学习，我们对众多的元字符进行了分类：

边界符

定义位置规则，必须用什么开头，用什么结尾



量词

定义重复次数规则

范围

表示字符的范围

字符类

区分各种字符，例如区分字母和数字

1.2 元字符

3. 范围：

表示字符的范围，定义的规则限定在某个范围，比如只能是英文字母，或者数字等等，用 [] 表示范围

[abc]	匹配包含的单个字符。也就是只有 a | b | c 这三个单字符返回true， 可以理解为多选1
[a-z]	连字符。来指定字符范围。[a-z] 表示 a 到 z 26个英文字母
[^abc]	取反符。[^a-z] 匹配除了小写字母以外的字符

1.2 元字符

3. 范围：

表示字符的分组和范围

- 使用连字符 - 表示一个范围

```
console.log(/^[a-z]$/.test('c'))  // true
```

- 比如：
  - [a-z] 表示 a 到 z 26个英文字母都可以
  - [a-zA-Z] 表示大小写都可以
  - [0-9] 表示 0~9 的数字都可以
- 认识下：

```
腾讯QQ号：  ^[1-9][0-9]{4,}$ (腾讯QQ号从10000开始)
```

总结

1. 请说下代码代表什么意思？
   - [abc] 匹配abc其中的任何单个字符
   - [a-z] 匹配26个小写英文字母其中的任何单个字符
   - [^a-z] 匹配除了26个小写英文字母之外的其他任何单个字符

案例

用户名验证案例

需求：用户名要求用户英文字母,数字,下划线或者短横线组成，并且用户名长度为 6~16 位

分析：

①：首先准备好这种正则表达式模式 /^[a-zA-Z0-9-_]{6,16}$/

②：当表单失去焦点就开始验证.

③：如果符合正则规范，则让后面的span标签添加 right 类

④：如果不符合正则规范，则让后面的span标签添加 wrong 类，同时显示提示框

练习

昵称案例（课堂作业）

需求：要求用户只能输入中文

分析：

①：首先准备好这种正则表达式模式　/^[\u4e00-\u9fa5]{2,8}$/

②：当表单失去焦点就开始验证.

③：如果符合正则规范，则让后面的span标签添加 right 类

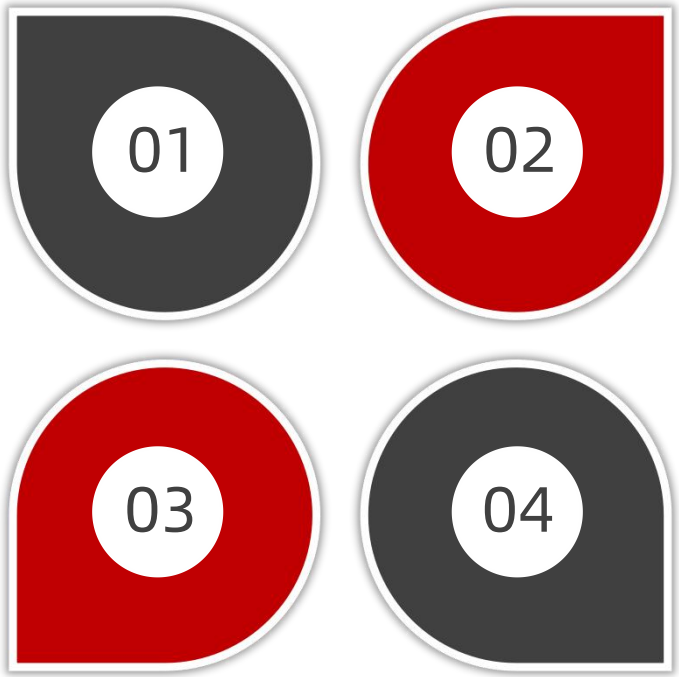
④：如果不符合正则规范，则让后面的span标签添加 wrong 类，同时显示提示框（只能输入中文）

1.2 元字符

- 为了方便记忆和学习，我们对众多的元字符进行了分类：

边界符

定义位置规则，必须用什么开头，用什么结尾



量词

定义重复次数规则

范围

表示字符的范围

字符类

区分各种字符，例如区分字母和数字

1.2 元字符

4. 字符类：

某些常见模式的简写方式，区分字母和数字

字符类	说明
\d	匹配0-9之间的任一数字，相当于[0-9]
\D	匹配所有0-9以外的字符，相当于 [^0-9]
\w	匹配任意的字母、数字和下划线，相当于[A-Za-z0-9_]
\W	除所有字母、数字和下划线以外的字符，相当于 [^A-Za-z0-9_]
\s	匹配空格（包括换行符、制表符、空格符等），相等于[\t\r\n\v\f]
\S	匹配非空格的字符，相当于 [^ \t\r\n\v\f]

```
日期格式： /^\d{4}-\d{1,2}-\d{1,2}$/
```

日期格式：2099-12-12

日期格式：2099-1-1

01

正则表达式

- 基本使用
- 元字符
- 替换和修饰符

- 目标：学习正则表达式概念及语法，编写简单的正则表达式实现字符的查找或检测。

1.3 替换和修饰符

- replace 替换方法，可以完成字符的替换
- 语法：

```
字符串.replace(/正则表达式/, '替换的文本')
```

1.3 替换和修饰符

- 修饰符约束正则执行的某些细节行为，如是否区分大小写、是否支持多行匹配等
- 语法：

```
/表达式/修饰符
```

- i 是单词 ignore 的缩写，正则匹配时字母不区分大小写
- g 是单词 global 的缩写，匹配所有满足正则表达式的结果

```
console.log(/a/i.test('a'))  // true
console.log(/a/i.test('A'))  // true
```

案例

利用正则隐藏手机号中间四位

分析：

①：手机号前三位显示，中间四位隐藏为****， 后面四位显示

②：把手机号利用正则利用小括号划分为三部分

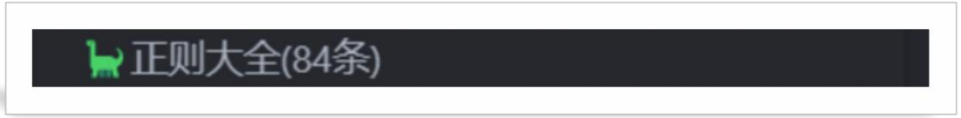
③：在replace中

$1 对应第一个小括号内容，$2 对应第2个括号内容，依次类推

④：字符串重复使用： 字符串.repeat(次数) 实现

推荐一个vscode正则插件

可以帮我们快速生成正则表达式 **any-rule**

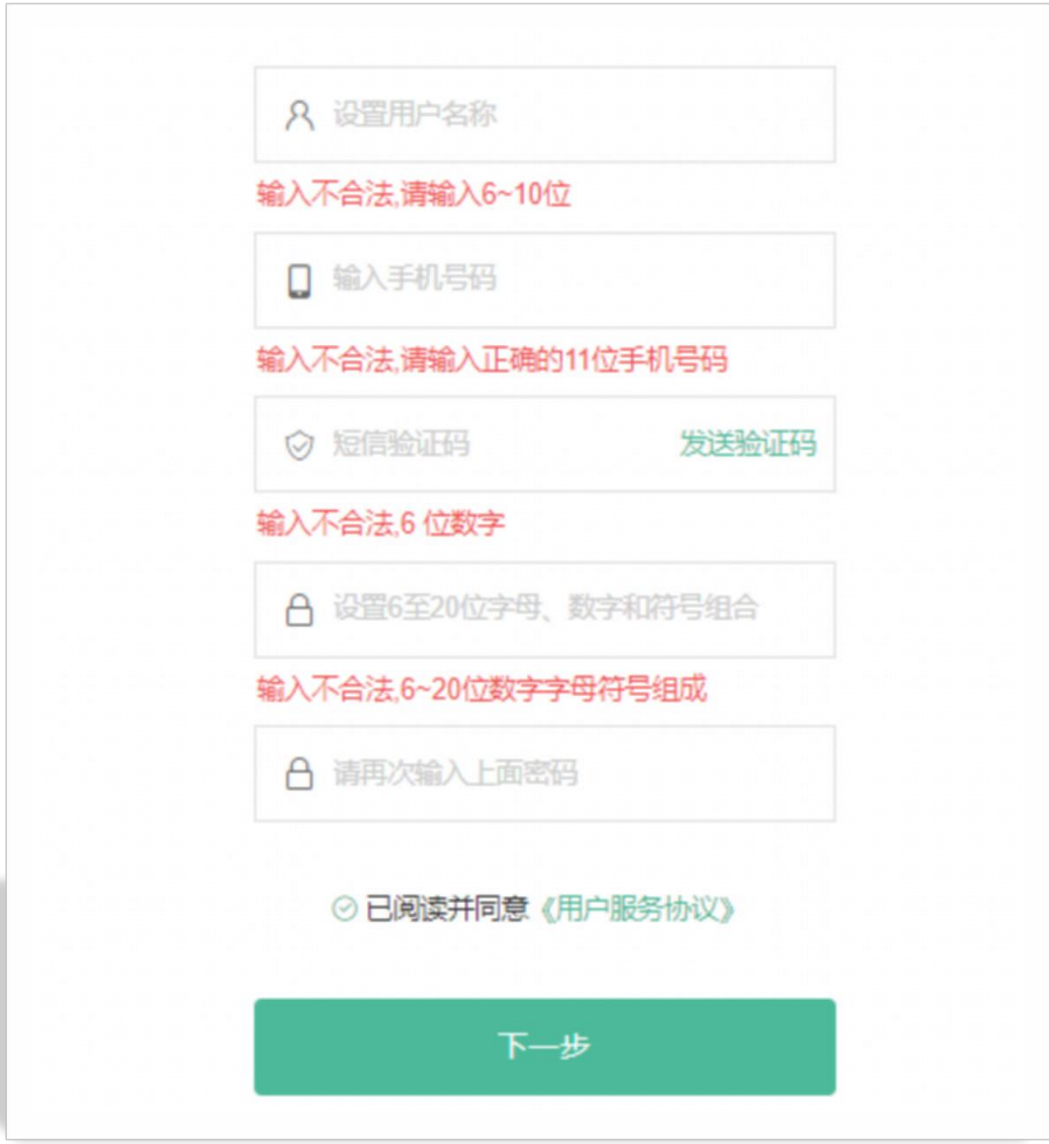


目录

Contents

案例

小兔鲜页面注册


设置用户名称
输入不合法,请输入6~10位
输入手机号码
输入不合法,请输入正确的11位手机号码
短信验证码
发送验证码
输入不合法,6 位数字
设置6至20位字母、数字和符号组合
输入不合法,6~20位数字字母符号组成
请再次输入上面密码
已阅读并同意《用户服务协议》
下一步

步骤 小兔鲜页面注册

分析业务：

- 发送验证码业务
- 表单验证业务
- 阅读同意业务
- 下一步验证业务

步骤 小兔鲜页面注册

分析业务：

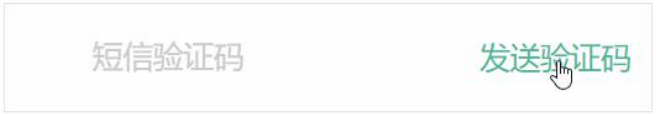
- 发送验证码业务
- 表单验证业务
- 阅读同意业务
- 下一步验证业务

步骤 小兔鲜页面注册

发送验证码业务

业务①： 发送验证码

- 用户点击之后，显示 ’05秒后重新获取’，
- 利用间歇函数 setInterval，做倒计时效果
- 时间到了，自动改为 重新获取



防止多次点击的问题：

思路：定义一个开关变量，防止代码多次执行 flag = true

1. 点击之后，代码执行一次之后，先关闭开关， flag = false 防止代码多次执行
2. 倒计时结束，时间到了，打开开关 flag = true ，可以进行下一次点击

步骤 小兔鲜页面注册

分析业务：

- 发送验证码业务
- 表单验证业务
- 阅读同意业务
- 下一步验证业务

步骤 小兔鲜页面注册

表单验证业务

业务②： 表单验证业务

1. **用户名验证**（注意封装函数 verifyName）

(1). 给input注册 change 事件，**值被修改并且失去焦点后触发**

(2). 正则规则： /^[a-zA-Z0-9-_]{6,10}$/

(3). 如果不符合要求，则出现 span 提示信息 并 return false 中断程序

(4). 如果正确则返回 return true ，之所以返回布尔值，是为了最后的提交做准备

步骤 小兔鲜页面注册

表单验证业务

业务②： 表单验证业务

2. **手机号验证**（注意封装函数 verifyPhone）

(1). 正则：/^1(3\d|4[5-9]|5[0-35-9]|6[567]|7[0-8]|8\d|9[0-35-9])\d{8}$/

(2). 其余同上

3. **验证码验证**（注意封装函数 verifyCode）

(1). 正则： /^\d{6}$/

(2). 其余同上

4. **密码验证**（注意封装函数 verifyPwd）

(1). 正则： /^[a-zA-Z0-9-_]{6,20}$/

(2). 其余同上

步骤 小兔鲜页面注册

表单验证业务

业务②： 表单验证业务

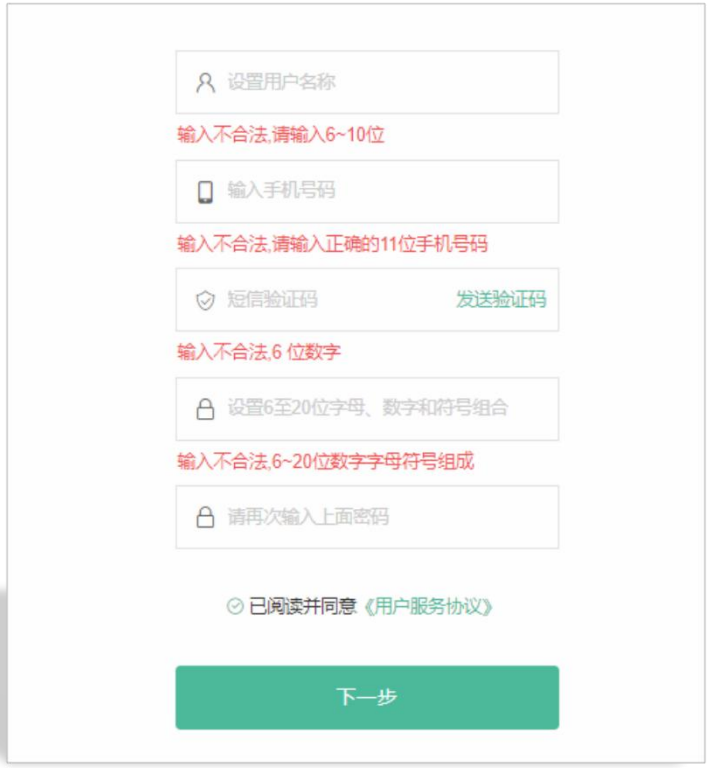
5. **再次密码验证**（注意封装函数 verifyConfirm）

(1). 如果再次密码的值不等于上面密码框的值则返回错误信息

(2). 其余同上

步骤 小兔鲜页面注册

分析业务：



- 发送验证码业务
- 表单验证业务
- 阅读同意业务
- 下一步验证业务

步骤 小兔鲜页面注册

阅读同意业务

业务③： 已阅读同意业务

1. 注册点击事件，可以 toggle 切换类实现不同样式

.icon-queren 是默认类没选中

.icon-queren2 则是选中样式

```
// 新增一个类名
对象.classList.add('类名')
// 移除一个类名
对象.classList.remove('类名')
// 切换一个类名
对象.classList.toggle('类名')
```

步骤 小兔鲜页面注册

分析业务：

- 发送验证码业务
- 表单验证业务
- 阅读同意业务
- 下一步验证业务

步骤 小兔鲜页面注册

下一步验证业务

需求④： 下一步提交验证业务

1. 使用 submit 提交事件
2. 如果没有 勾选同意协议，则提示 ’需要勾选’，并且 要阻止提交行为

元素.classList.contains() 看看有没有包含某个类，如果有则返回true，么有则返回false

```
// 添加类名
元素.classList.add('类名')
// 删除类名
元素.classList.remove('类名')
// 切换类名
元素.classList.toggle('类名')
// 判断是否包含某个类名  有返回true，没有返回false
元素.classList.contains('类名')
```

3. 依次判断上面的表单是否通过，只要有一个没有通过的就阻止

目录

Contents

案例 小兔鲜登录页面

- tab栏切换业务
- 登录业务

案例 小兔鲜登录页面

业务①： tab切换

关键词： 事件委托，排他思想

案例 小兔鲜登录页面

业务①： tab切换 总结方案

关键词： 事件委托，排他思想

方案1：

1. 上盒子添加类，删除类
2. 下盒子通过类 active 控制显示隐藏

排他思想：

(1) 先移除 active 类，就隐藏

(2) 对应索引号的下盒子添加 active 类就显示

方案2：

1. 上盒子添加类，删除类
2. 下盒子通过循环全部隐藏（暴力写法）

排他思想：

(1) 先利用循环把全部下盒子隐藏

(2) 对应索引号的下盒子添加 display：block 显示

案例 小兔鲜登录页面

业务②： 点击登录可以跳转页面

1. 注册表单提交事件，阻止默认行为
2. 如果没有勾选同意，则提示要勾选
3. required 属性可以让表单不能为空
4. 假设登录成功

   把用户名记录到本地存储中

   同时跳转到首页 location.href

案例 小兔鲜首页页面

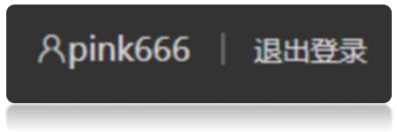
需求：

1. 从登录页面跳转过来之后，自动显示用户名
2. 如果点击退出，则不显示用户名

案例 小兔鲜首页页面

核心思路：判断本地存储有没有数据（用户名）

如果有就显示 用户名和退出登录



如果没有则显示请先登录和免费注册

已登录标签格式：

```
<li>
  <a href="javascript:;">
    <i class="iconfont icon-user">pink666</i>
  </a>
</li>
<li><a href="javascript:;">退出登录</a></li>
```

未登录标签格式：(默认结构)

```
<li><a href="./login.html">请先登录</a></li>
<li><a href="./register.html">免费注册</a></li>
```

案例 小兔鲜首页页面

```
<li>
  <a href="javascript:;">
    <i class="iconfont icon-user">pink666</i>
  </a>
</li>
<li><a href="javascript:;">退出登录</a></li>
```

步骤：

①. render 渲染函数，因为一会的退出还需要用到

(1) 读取本地存储数据，如果有数据，第一个和第二个li按照结构渲染

(2) 如果本地存储没有数据，则复原为默认的结构

②： 点击退出登录

(1) 删除本地存储对应的用户名数据

(2) 重新调用渲染函数即可

```
<li><a href="./login.html">请先登录</a></li>
<li><a href="./register.html">免费注册</a></li>
```

案例

小兔鲜首页页面

需求：

如果首页是移动端打开的，则自动跳到移动端页面

比如 m.itcast.cn 上去

提示： 利用 BOM 的 navigator

传智教育旗下高端IT教育品牌