

[Home](#) > [Regular Expressions](#) > [Java Regex : Validate International Phone Numbers](#)

Java Regex : Validate International Phone Numbers

November 12, 2014 by Lokesh Gupta

In this regex tutorial, we will learn to validate international phone numbers based on industry-standard notation specified by [ITU-T E.123](#)

The rules and conventions used to print international phone numbers vary significantly around the world, so it's hard to provide meaningful validation for an international phone number unless you adopt a strict format. Fortunately, there is a simple, industry-standard notation specified by ITU-T E.123. This notation requires that international phone numbers include a leading plus sign (known as the international prefix symbol), and allows only spaces to separate groups of digits.

Also thanks to the international phone numbering plan ([ITU-T E.164](#)), phone numbers cannot contain more than 15 digits. The shortest international phone numbers in use contain seven digits.

Using Regex to Validate International Phone Numbers

Regex : `^\+(?:[0-9] ?){6,14}[0-9]$`

`^` # Assert position at the beginning of the string.
`\+` # Match a literal "+" character.
`(?:` # Group but don't capture:
`[0-9]` # Match a digit.
`\s` # Match a space character
`?` # between zero and one time.
`)` # End the noncapturing group.
`{6,14}` # Repeat the group between 6 and 14 times.
`[0-9]` # Match a digit.
`$` # Assert position at the end of the string.

Above regular expression can be used to validate international phone numbers based on ITU-T standards. Let's look at one example.

```
List phoneNumbers = new ArrayList();
phoneNumbers.add("+1 1234567890123");
phoneNumbers.add("+12 123456789");
phoneNumbers.add("+123 123456");

String regex = "^\+(?:[0-9] ?){6,14}[0-9]";

Pattern pattern = Pattern.compile(regex);

for(String email : phoneNumbers)
{
```

```
Matcher matcher = pattern.matcher(email);
System.out.println(email + " : "+ matcher.matches());
}
```

Output:

```
+1 1234567890123 : true
+12 123456789 : true
+123 123456 : true
```

Validate international phone numbers in EPP format

This regular expression follows the international phone number notation specified by the [Extensible Provisioning Protocol](#) (EPP). EPP is a relatively recent protocol (finalized in 2004), designed for communication between domain name registries and registrars. It is used by a growing number of domain name registries, including .com, .info, .net, .org, and .us. The significance of this is that EPP-style international phone numbers are increasingly used and recognized, and therefore provide a good alternative format for storing (and validating) international phone numbers.

EPP-style phone numbers use the format +**CCC.NNNNNNNNNN**x**EEEE**, where C is the 1–3 digit country code, N is up to 14 digits, and E is the (optional) extension. The leading plus sign and the dot following the country code are required. The literal “x” character is required only if an extension is provided.

Regex : `^\+[0-9]{1,3}\.[0-9]{4,14}(?:x.+)?`

```
List phoneNumbers = new ArrayList();
phoneNumbers.add("+123.123456x4444");
phoneNumbers.add("+12.1234x11");
phoneNumbers.add("+1.123456789012x123456789");
```

```
String regex = "^\\+[0-9]{1,3}\\.[0-9]{4,14}(?:x.+)?$";

Pattern pattern = Pattern.compile(regex);

for(String email : phoneNumbers)
{
    Matcher matcher = pattern.matcher(email);
    System.out.println(email + " : "+ matcher.matches());
}
```

Output:

```
+123.123456x4444 : true
+12.1234x11 : true
+1.123456789012x123456789 : true
```

Feel free to edit above regex and play with it to match more strict phone number formats, you have in your mind.

Happy Learning !!

Share this:

Facebook



LinkedIn



Reddit



Twitter



G+ Google



WhatsApp

Readers also found them useful:

[Java Regex : Validate Credit Card Numbers](#)

[Java Regex : Match Any Character in "Greek Extended" or Greek script](#)

[Java Regex : Validate and Format North American Phone Numbers](#)

[Java Regex – Password Validation Example](#)

[Java Regex : Validate / Limit the number of words in input](#)

[Java Regex : Validate U.K. Postal Codes \(Postcodes\)](#)

[Java Regex : Validate US Postal Zip Codes](#)

[Java Regular Expressions : Meta Characters](#)

Java Regex – Match Word with All Misspellings

Java Regex : Validate Canadian Postal Zip Codes

Subscribe to Blog via Email

Enter your email address to subscribe to this blog and receive notifications of new posts by email.

Join 3,873 other subscribers

SUBSCRIBE

About Lokesh Gupta

Founded HowToDoInJava.com in late 2012. I love computers, programming and solving problems everyday. A family guy with fun loving nature. You can find me on [Facebook](#), [Twitter](#) and [Google Plus](#).

Want to ask any question? Or suggest anything?


Enter your comment here...

Search Tutorials

Type and Press ENTER...



Lokesh Gupta

 **Seguir**

4.844 seguidores



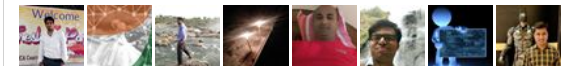
How To Do In Java

14.810 curtidas

Curtir Página

Fale conosco

Seja o primeiro de seus amigos a curtir isso.



Recommended

[10 Life Lessons](#)

[Generate Secure Passwords](#)

[How Web Servers work?](#)

[How Java I/O Works Internally?](#)

[5 Class Design Principles](#)

[Unit Testing Best Practices](#)

[Exception Handling Best Practices](#)

[Best Way to Learn Java](#)

[Java Best Practices](#)

[Java Interview Questions](#)

[Best Java Books](#)

[REST API Tutorial](#)

Advertisements

Most Viewed Today

[Spring @Component, @Repository, @Service and @Controller Annotations](#)

[Spring MVC Interview Questions with Answers](#)

[Spring RESTful Client - RestTemplate Example](#)

[Java 8 Predicate Examples](#)

[Reading/writing Excel Files in Java : POI tutorial](#)

[Top Spring Core Interview Questions with Answers](#)

[5 Class Design Principles \[S.O.L.I.D.\] in Java](#)

[How to Work With wait\(\), notify\(\) and notifyAll\(\) in Java?](#)

[A Guide to Object Cloning in Java](#)

[Generate Secure Password Hash : MD5, SHA, PBKDF2, BCrypt Examples](#)

21/04/2017

Java Regex : Validate International Phone Numbers - HowToDoInJava

[JSON Formatter and Minifier](#)

[XML Formatter and Minifier](#)

[CSS Formatter and Minifier](#)

[HTML Formatter and Minifier](#)

[Advertise](#)

[Contact Us](#)

[Privacy policy](#)

[About Me](#)

[Online tests](#)

References

[Java 6 API](#)

[Java 7 API](#)

[Java 8 API](#)

[Spring 3 Reference](#)

[Spring 4 References](#)

[RESTEasy 3.1.x](#)

[Hibernate User Guide](#)

[JUnit Wiki](#)

[Maven FAQs](#)

Copyright © 2016 · HowToDoInJava.com · All Rights Reserved