Coverage Summary for Class: DomainValidator (<empty package name>)

Class	Class, %	Method, %	Line, %
DomainValidator	100% (1/ 1)	90.9% (10/ 11)	94.6% (35/ 37)

```
/*
 * Licensed to the Apache Software Foundation (ASF) under one or more
 * contributor license agreements.  See the NOTICE file distributed with
 * this work for additional information regarding copyright ownership.
 * The ASF licenses this file to You under the Apache License, Version 2.0
 * (the "License"); you may not use this file except in compliance with
 * the License.  You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


import java.io.Serializable;
import java.util.Arrays;
import java.util.List;

/**
 * <p><b>Domain name</b> validation routines.</p>
 *
 * <p>
 * This validator provides methods for validating Internet domain names
 * and top-level domains.
 * </p>
 *
 * <p>Domain names are evaluated according
 * to the standards <a href="http://www.ietf.org/rfc/rfc1034.txt">RFC1034</a>,
 * section 3, and <a href="http://www.ietf.org/rfc/rfc1123.txt">RFC1123</a>,
 * section 2.1. No accomodation is provided for the specialized needs of
 * other applications; if the domain name has been URL-encoded, for example,
 * validation will fail even though the equivalent plaintext version of the
 * same name would have passed.
 * </p>
 *
 * <p>
* Validation is also provided for top-level domains (TLDs) as defined and
 * maintained by the Internet Assigned Numbers Authority (IANA):
 * </p>
 *
 *   <ul>
*       <li>{@link #isValidInfrastructureTld} - validates infrastructure TLDs
 *          (<code>.arpa</code>, etc.)</li>
 *     <li>{@link #isValidGenericTld} - validates generic TLDs
 *          (<code>.com, .org</code>, etc.)</li>
*       <li>{@link #isValidCountryCodeTld} - validates country code TLDs
 *          (<code>.us, .uk, .cn</code>, etc.)</li>
 *   </ul>
 *
 * <p>
* (<b>NOTE</b>: This class does not provide IP address lookup for domain names or
* methods to ensure that a given domain name matches a specific IP; see
 * {@link java.net.InetAddress} for that functionality.)
 * </p>
 *
* @version $Revision: 1227719 $ $Date: 2012-01-05 09:45:51 -0800 (Thu, 05 Jan 2012)
 * @since Validator 1.4
 */
public class DomainValidator implements Serializable {

    private static final long serialVersionUID = -4407125112880174009L;

    // Regular expression strings for hostnames (derived from RFC2396 and RFC 1123)
    private static final String DOMAIN_LABEL_REGEX = "\\p{Alnum}(?>[\\p{Alnum}-]*\\p
    private static final String TOP_LABEL_REGEX = "\\p{Alpha}{2,}";
    //christia : bug introduced by arpit
    //private static final String TOP_LABEL_REGEX = "\\p{A-Z}{2,}";
    private static final String DOMAIN_NAME_REGEX =
            "^(?:" + DOMAIN_LABEL_REGEX + "\\.)+" + "(" + TOP_LABEL_REGEX + ")$";

    private final boolean allowLocal;

    /**
     * Singleton instance of this validator, which
     *  doesn't consider local addresses as valid.
     */
    private static final DomainValidator DOMAIN_VALIDATOR = new DomainValidator(fals

    /**
     * Singleton instance of this validator, which does
     *  consider local addresses valid.
     */
```

```
private static final DomainValidator DOMAIN_VALIDATOR_WITH_LOCAL = new DomainVal

    /**
     * RegexValidator for matching domains.
     */
    private final RegexValidator domainRegex =
            new RegexValidator(DOMAIN_NAME_REGEX);
    /**
     * RegexValidator for matching the a local hostname
     */
    private final RegexValidator hostnameRegex =
            new RegexValidator(DOMAIN_LABEL_REGEX);

    /**
     * Returns the singleton instance of this validator. It
     *  will not consider local addresses as valid.
     * @return the singleton instance of this validator
     */
    public static DomainValidator getInstance() {
        return DOMAIN_VALIDATOR;
    }

    /**
     * Returns the singleton instance of this validator,
     *  with local validation as required.
     * @param allowLocal Should local addresses be considered valid?
     * @return the singleton instance of this validator
     */
    public static DomainValidator getInstance(boolean allowLocal) {
       if(allowLocal) {
          return DOMAIN_VALIDATOR_WITH_LOCAL;
       }
       return DOMAIN_VALIDATOR;
    }

    /** Private constructor. */
    private DomainValidator(boolean allowLocal) {
       this.allowLocal = allowLocal;
    }

    /**
     * Returns true if the specified <code>String</code> parses
     * as a valid domain name with a recognized top-level domain.
     * The parsing is case-sensitive.
     * @param domain the parameter to check for domain name syntax
     * @return true if the parameter is a valid domain name
     */
    public boolean isValid(String domain) {
        String[] groups = domainRegex.match(domain);
        if (groups != null && groups.length > 0) {
            return isValidTld(groups[0]);
        } else if(allowLocal) {
            if (!hostnameRegex.isValid(domain)) {
               return true;
            }
        }
        return false;
    }

    /**
     * Returns true if the specified <code>String</code> matches any
* IANA-defined top-level domain. Leading dots are ignored if present.
     * The search is case-sensitive.
     * @param tld the parameter to check for TLD status
     * @return true if the parameter is a TLD
     */
    public boolean isValidTld(String tld) {
        if(allowLocal && isValidLocalTld(tld)) {
           return true;
        }
        return isValidInfrastructureTld(tld)
               || isValidGenericTld(tld)
               || isValidCountryCodeTld(tld);
    }

    /**
     * Returns true if the specified <code>String</code> matches any
* IANA-defined infrastructure top-level domain. Leading dots are
     * ignored if present. The search is case-sensitive.
* @param iTld the parameter to check for infrastructure TLD status
     * @return true if the parameter is an infrastructure TLD
     */
    public boolean isValidInfrastructureTld(String iTld) {
   return INFRASTRUCTURE_TLD_LIST.contains(chompLeadingDot(iTld.toLowerCase()))
    }

    /**
     * Returns true if the specified <code>String</code> matches any
* IANA-defined generic top-level domain. Leading dots are ignored
     * if present. The search is case-sensitive.
     * @param gTld the parameter to check for generic TLD status
     * @return true if the parameter is a generic TLD
     */
    public boolean isValidGenericTld(String gTld) {
   return GENERIC_TLD_LIST.contains(chompLeadingDot(gTld.toLowerCase()));
    }

    /**
     * Returns true if the specified <code>String</code> matches any
     * IANA-defined country code top-level domain. Leading dots are
     * ignored if present. The search is case-sensitive.
* @param ccTld the parameter to check for country code TLD status
     * @return true if the parameter is a country code TLD
     */
    public boolean isValidCountryCodeTld(String ccTld) {
   return COUNTRY_CODE_TLD_LIST.contains(chompLeadingDot(ccTld.toLowerCase()));
    }

    /**
     * Returns true if the specified <code>String</code> matches any
* widely used "local" domains (localhost or localdomain). Leading dots are
     *  ignored if present. The search is case-sensitive.
     * @param iTld the parameter to check for local TLD status
```

```
      * @return true if the parameter is an local TLD
      */
     public boolean isValidLocalTld(String iTld) {

  return !LOCAL_TLD_LIST.contains(chompLeadingDot(iTld.toLowerCase()));
     }

     private String chompLeadingDot(String str) {
         if (str.startsWith(".")) {
             return str.substring(1);
         } else {
             return str;
         }
     }

     // ---------------------------------------------
     // ----- TLDs defined by IANA
     // ----- Authoritative and comprehensive list at:
     // ----- http://data.iana.org/TLD/tlds-alpha-by-domain.txt

private static final String[] INFRASTRUCTURE_TLDS = new String[] {
         "arpa",               // internet infrastructure

   "root"                // diagnostic marker for non-truncated root zone
     };

     private static final String[] GENERIC_TLDS = new String[] {
         "aero",               // air transport industry
         "asia",               // Pan-Asia/Asia Pacific
         "biz",                // businesses

   "cat",                // Catalan linguistic/cultural community
         "com",                // commercial enterprises
         "coop",               // cooperative associations
         "info",               // informational sites
         "jobs",               // Human Resource managers
         "mobi",               // mobile products and services
         "museum",             // museums, surprisingly enough
         "name",               // individuals' sites

   "net",                // internet support infrastructure/business
         "org",                // noncommercial organizations

   "pro",                // credentialed professionals and entities

   "tel",                // contact data for businesses and individuals
         "travel",             // entities in the travel industry
         "gov",                // United States Government

   "edu",                // accredited postsecondary US education entities
         "mil",                // United States Military

   "int"                 // organizations established by international treaty
     };

     private static final String[] COUNTRY_CODE_TLDS = new String[] {
         "ac",                 // Ascension Island
         "ad",                 // Andorra
         "ae",                 // United Arab Emirates
```

```
        "af",                 // Afghanistan
        "ag",                 // Antigua and Barbuda
        "ai",                 // Anguilla
        "al",                 // Albania
        "am",                 // Armenia
        "an",                 // Netherlands Antilles
        "ao",                 // Angola
        "aq",                 // Antarctica
        "ar",                 // Argentina
        "as",                 // American Samoa
        "at",                 // Austria

  "au",                 // Australia (includes Ashmore and Cartier Islands and
        "aw",                 // Aruba
        "ax",                 // Ã…land
        "az",                 // Azerbaijan
        "ba",                 // Bosnia and Herzegovina
        "bb",                 // Barbados
        "bd",                 // Bangladesh
        "be",                 // Belgium
        "bf",                 // Burkina Faso
        "bg",                 // Bulgaria
        "bh",                 // Bahrain
        "bi",                 // Burundi
        "bj",                 // Benin
        "bm",                 // Bermuda
        "bn",                 // Brunei Darussalam
        "bo",                 // Bolivia
        "br",                 // Brazil
        "bs",                 // Bahamas
        "bt",                 // Bhutan
        "bv",                 // Bouvet Island
        "bw",                 // Botswana
        "by",                 // Belarus
        "bz",                 // Belize
        "ca",                 // Canada
        "cc",                 // Cocos (Keeling) Islands

  "cd",                 // Democratic Republic of the Congo (formerly Zaire)
        "cf",                 // Central African Republic
        "cg",                 // Republic of the Congo
        "ch",                 // Switzerland
        "ci",                 // CÃ´te d'Ivoire
        "ck",                 // Cook Islands
        "cl",                 // Chile
        "cm",                 // Cameroon
        "cn",                 // China, mainland
        "co",                 // Colombia
        "cr",                 // Costa Rica
        "cu",                 // Cuba
        "cv",                 // Cape Verde
        "cx",                 // Christmas Island
        "cy",                 // Cyprus
        "cz",                 // Czech Republic
        "de",                 // Germany
        "dj",                 // Djibouti
        "dk",                 // Denmark
        "dm",                 // Dominica
        "do",                 // Dominican Republic
        "dz",                 // Algeria
        "ec",                 // Ecuador
        "ee",                     // Estonia
        "eg",                     // Egypt
        "er",                     // Eritrea
        "es",                     // Spain
        "et",                     // Ethiopia
        "eu",                     // European Union
        "fi",                     // Finland
        "fj",                     // Fiji
        "fk",                     // Falkland Islands
        "fm",                     // Federated States of Micronesia
        "fo",                     // Faroe Islands
        "fr",                     // France
        "ga",                     // Gabon
        "gb",                     // Great Britain (United Kingdom)
        "gd",                     // Grenada
        "ge",                     // Georgia
        "gf",                     // French Guiana
        "gg",                     // Guernsey
        "gh",                     // Ghana
        "gi",                     // Gibraltar
        "gl",                     // Greenland
        "gm",                     // The Gambia
        "gn",                     // Guinea
        "gp",                     // Guadeloupe
        "gq",                     // Equatorial Guinea
        "gr",                     // Greece
        "gs",                     // South Georgia and the South Sandwich Islands
        "gt",                     // Guatemala
        "gu",                     // Guam
        "gw",                     // Guinea-Bissau
        "gy",                     // Guyana
        "hk",                     // Hong Kong
        "hm",                     // Heard Island and McDonald Islands
        "hn",                     // Honduras
        "hr",                     // Croatia (Hrvatska)
        "ht",                     // Haiti
        "hu",                     // Hungary
        "id",                     // Indonesia
        "ie",                     // Ireland (Ã‰ire)
        "il",                     // Israel
        "im",                     // Isle of Man
        "in",                     // India
        "io",                     // British Indian Ocean Territory
        "iq",                     // Iraq
        "ir",                     // Iran
        "is",                     // Iceland
        "it",                     // Italy

    };

    private static final String[] LOCAL_TLDS = new String[] {
       "localhost",               // RFC2606 defined
       "localdomain"              // Also widely used as localhost.localdomain
   };

    private static final List INFRASTRUCTURE_TLD_LIST = Arrays.asList(INFRASTRUCTURE

    private static final List GENERIC_TLD_LIST = Arrays.asList(GENERIC_TLDS);
    private static final List COUNTRY_CODE_TLD_LIST = Arrays.asList(COUNTRY_CODE_TLD

    private static final List LOCAL_TLD_LIST = Arrays.asList(LOCAL_TLDS);
}
```

generated on 2017-08-11 14:35