



链滴

Java 正则提取字符串中的 URL 链接

作者: [pencilso](#)

原文链接: <https://ld246.com/article/1533197881404>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

提取URL链接

```
public static void main(String[] args) {  
    String data = "#在抖音，记录美好生活#这大概就是冰雪美人吧..... http://v.douyin.com/eUWYt  
    / 复制此链接，打开【抖音短视频】，直接观看视频！";  
    Matcher matcher = Patterns.WEB_URL.matcher(data);  
    if (matcher.find()) {  
        System.out.println(matcher.group());  
    }  
}  
#最后输出结果为  
http://v.douyin.com/eUWYth
```

工具类

该工具类 摘自android.util包下

```
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
/**  
 * Commonly used regular expression patterns.  
 */  
public class Patterns {  
    /**  
     * Regular expression to match all IANA top-level domains.  
     * List accurate as of 2011/07/18. List taken from:  
     * http://data.iana.org/TLD/tlds-alpha-by-domain.txt  
     * This pattern is auto-generated by frameworks/ex/common/tools/make-iana-tld-pattern.  
     *  
     * @deprecated Due to the recent proliferation of gTLDs, this API is  
     * expected to become out-of-date very quickly. Therefore it is now  
     * deprecated.  
     */  
    @Deprecated  
    public static final String TOP_LEVEL_DOMAIN_STR =  
        "((aero|arpa|asia|a[cdefgilmnoqrstuwxz])"  
        + "|(biz|b[abdefghijmnorstuvwxyz])"  
        + "|(cat|com|coop|c[acdfghiklmnoruvxyz])"  
        + "|d[ejkmoz]"  
        + "|(edu|e[cegrstu])"  
        + "|f[ijkmor]"  
        + "|(gov|g[abdefghilmnpqrstuwy])"  
        + "|h[kmnrtu]"  
        + "|(info|int|i[delmnoqrst])"  
        + "|(jobs|j[emop])"  
        + "|k[eghimnprwyz]"  
        + "|l[abcikrstuvy]"  
        + "|(mil|mobi|museum|m[acdeghklmnopqrstuvwxyz])"  
        + "|(name|net|n[acefgilopruz])"
```

```

+ "|(org|om)"
+ "|(pro|p[aefghklmnrstwy])"
+ "|qa"
+ "|r[eosuw]"
+ "|s[abcdefghijklmnortuvwxyz]"
+ "|(tel|travel|t[cdfghjklmnoprtvwz])"
+ "|u[agksyz]"
+ "|v[aceginu]"
+ "|w[fs]"
+ "|(\u03b4\u03bf\u03ba\u03b9\u03bc\u03ae|\u0438\u0441\u043f\u044b\u0442\u0430
u043d\u0438\u0435|\u0440\u0444|\u0441\u0440\u0431|\u05d8\u05e2\u05e1\u05d8|\u0622
u0632\u0645\u0627\u06cc\u0634\u06cc|\u0625\u062e\u062a\u0628\u0627\u0631|\u0627\u
644\u0627\u0631\u062f\u0646|\u0627\u0644\u062c\u0632\u0627\u0626\u0631|\u0627\u06
4\u0633\u0639\u0648\u062f\u064a\u0629|\u0627\u0644\u0645\u063a\u0631\u0628|\u0627
u0645\u0627\u0631\u0627\u062a|\u0628\u06be\u0627\u0631\u062a|\u062a\u0648\u0646\
0633|\u0633\u0648\u0631\u064a\u0629|\u0641\u0644\u0633\u0637\u064a\u0646|\u0642\u
637\u0631|\u0645\u0635\u0631|\u092a\u0930\u0940\u0915\u094d\u0937\u093e|\u092d\u0
3e\u0930\u0924|\u09ad\u09be\u09b0\u09a4|\u0a2d\u0a3e\u0a30\u0a24|\u0aad\u0abe\u0a
0\u0aa4|\u0b87\u0ba8\u0bcd\u0ba4\u0bbf\u0baf\u0bbe|\u0b87\u0bb2\u0b99\u0bcd\u0b9
\u0bc8|\u0b9a\u0bbf\u0b99\u0bcd\u0b95\u0baa\u0bcd\u0baa\u0bc2\u0bb0\u0bcd|\u0baa
u0bb0\u0bbf\u0b9f\u0bcd\u0b9a\u0bc8|\u0c2d\u0c3e\u0c30\u0c24\u0c4d|\u0dbd\u0d82\u
d9a\u0dcf|\u0e44\u0e17\u0e22|\u30c6\u30b9\u30c8|\u4e2d\u56fd|\u4e2d\u570b|\u53f0\u6
7e|\u53f0\u7063|\u65b0\u52a0\u5761|\u6d4b\u8bd5|\u6e2c\u8a66|\u9999\u6e2f|\ud14c\uc
a4\ud2b8|\ud55c\ud6d|xn\|\-\|-0zwm56d|xn\|\-\|-11b5bs3a9aj6g|xn\|\-\|-3e0b707e|xn\|\-\|-4
brj9c|xn\|\-\|-80akhbyknj4f|xn\|\-\|-90a3ac|xn\|\-\|-9t4b11yi5a|xn\|\-\|-clchc0ea0b2g2a9gcd|xn
\|\-\|-deba0ad|xn\|\-\|-fiqs8s|xn\|\-\|-fiqz9s|xn\|\-\|-fpcrj9c3d|xn\|\-\|-fzc2c9e2c|xn\|\-\|-g6w251d
xn\|\-\|-gecrj9c|xn\|\-\|-h2brj9c|xn\|\-\|-hgbk6aj7f53bba|xn\|\-\|-hlcj6aya9esc7a|xn\|\-\|-j6w193
|xn\|\-\|-jxalpdlp|xn\|\-\|-kgbechtv|xn\|\-\|-kprw13d|xn\|\-\|-kqry57d|xn\|\-\|-lgbbat1ad8j|xn\|\-\|
-mgbaam7a8h|xn\|\-\|-mgbayh7gpa|xn\|\-\|-mgbbh1a71e|xn\|\-\|-mgbc0a9azcg|xn\|\-\|-mgbe
p4a5d4ar|xn\|\-\|-o3cw4h|xn\|\-\|-ogbpf8fl|xn\|\-\|-p1ai|xn\|\-\|-pgbs0dh|xn\|\-\|-s9brj9c|xn\|\-\|
-wgbh1c|xn\|\-\|-wgbl6a|xn\|\-\|-xkc2a13hye2a|xn\|\-\|-xkc2d13a5ee0h|xn\|\-\|-yfro4i67o|xn\|\-\|
ygb2ammx|xn\|\-\|-zckzah|xxx)"
+ "|y[et]"
+ "|z[amw])";
}

/**
 * Regular expression pattern to match all IANA top-level domains.
 * @deprecated This API is deprecated. See {@link #TOP_LEVEL_DOMAIN_STR}.
 */
@Deprecated
public static final Pattern TOP_LEVEL_DOMAIN =
    Pattern.compile(TOP_LEVEL_DOMAIN_STR);

/**
 * Regular expression to match all IANA top-level domains for WEB_URL.
 * List accurate as of 2011/07/18. List taken from:
 * http://data.iana.org/TLD/tlds-alpha-by-domain.txt
 * This pattern is auto-generated by frameworks/ex/common/tools/make-iana-tld-pattern.
y
*
* @deprecated This API is deprecated. See {@link #TOP_LEVEL_DOMAIN_STR}.
*/
@Deprecated
public static final String TOP_LEVEL_DOMAIN_STR_FOR_WEB_URL =

```

```

"(?:" +
+ "(?:aero|arpa|asia|a[cdefgilmnoqrstuwxz])"
+ "|(?:biz|b[abdefghijmnorstuvwxyz])"
+ "|(?:cat|com|coop|c[acdfghiklmnoruvwxyz])"
+ "|d[ejkmoz]"
+ "|(?:edu|e[cegrstu])"
+ "|f[ijkmor]"
+ "|(?:gov|g[abdefghilmnpqrstuwy])"
+ "|h[kmnrtu]"
+ "|(?:info|int|i[delmnoqrst])"
+ "|(?:jobs|j[emop])"
+ "|k[eghimnprwyz]"
+ "|l[abcikrstuvy]"
+ "|(?:mil|mobi|museum|m[acdeghklmnopqrstuvwxyz])"
+ "|(?:name|net|n[acefgilopruz])"
+ "|(?:org|om)"
+ "|(?:pro|p[aefghklmnrstwy])"
+ "|qa"
+ "|r[eosuw]"
+ "|s[abcdeghijklnortuvwxyz]"
+ "|(?:tel|travel|t[cdfghjklmnoprtvwz])"
+ "|u[agksyz]"
+ "|v[aceginu]"
+ "|w[fs]"
+ "|(?:\u03b4\u03bf\u03ba\u03b9\u03bc\u03ae|\u0438\u0441\u043f\u044b\u0442\u040d\u043d\u0438\u0435\u0440\u0444|\u0441\u0440\u0431|\u05d8\u05e2\u05e1\u05d8\u062\u0632\u0645\u0627\u06cc\u0634\u06cc|\u0625\u062e\u062a\u0628\u0627\u0631|\u0627\u0644\u0627\u0631\u062f\u0646|\u0627\u0644\u062c\u0632\u0627\u0626\u0631|\u0627\u0644\u0633\u0639\u0648\u062f\u064a\u0629|\u0627\u0644\u0645\u063a\u0631\u0628|\u067\u0645\u0627\u0631\u0627\u062a|\u0628\u06be|\u0627\u0631\u062a|\u062a\u0648\u0633\u0631\u064a\u0629|\u0641\u0644\u0633\u0637\u064a\u0646|\u0642\u0637\u0631|\u0645\u0635\u0631|\u092a\u0930\u0940\u0915\u094d\u0937\u093e|\u092d\u093e\u0930\u0924|\u09ad\u09be|\u09b0\u09a4|\u0a2d\u0a3e\u0a30\u0a24|\u0aad\u0abe|\u0b0\u0aa4|\u0b87\u0ba8\u0bcd\u0ba4\u0bbf\u0baf\u0bbe|\u0b87\u0bb2\u0b99\u0bcd\u0b5\u0bc8\u0b9a\u0bbf\u0b99\u0bcd\u0b95\u0baa\u0bcd\u0bba\u0bc2\u0bb0\u0bcd|\u0ba\u0bb0\u0bbf\u0b9f\u0bcd\u0b9a\u0bc8|\u0c2d\u0c3e\u0c30\u0c24\u0c4d|\u0dbd\u0d82|\u0d9a\u0dcf|\u0e44\u0e17\u0e22|\u30c6\u30b9\u30c8|\u4e2d\u5fd|\u4e2d\u570b|\u53f0\u0e7e|\u53f0\u7063|\u65b0\u52a0\u5761|\u6d4b\u8bd5|\u6e2c\u8a66|\u9999\u6e2f|\u0d14c\u2a4\u0d2b8|\u0d55c\uad6d|xn\\-\u0zwmp56d|xn\\-11b5bs3a9aj6g|xn\\-3e0b707e|xn\\-5brj9c|xn\\-80akhbyknj4f|xn\\-90a3ac|xn\\-9t4b11yi5a|xn\\-clchc0ea0b2g2a9gcd|xn\\-deba0ad|xn\\-fiqs8s|xn\\-fiqz9s|xn\\-fpcrj9c3d|xn\\-fzc2c9e2c|xn\\-g6w25d|xn\\-gecrj9c|xn\\-h2brj9c|xn\\-hgbk6aj7f53bba|xn\\-hlcj6aya9esc7a|xn\\-j6w13g|xn\\-jxalpd|p|xn\\-kgbechtv|xn\\-kprw13d|xn\\-kpry57d|xn\\-lgbbat1ad8j|xn\\-mgbbaam7a8h|xn\\-mgbayh7gpa|xn\\-mgbbh1a71e|xn\\-mgbcoa9azcg|xn\\-berp4a5d4ar|xn\\-o3cw4h|xn\\-ogbpffl|xn\\-p1ai|xn\\-pgbs0dh|xn\\-s9brj9c|xn\\-wgbh1c|xn\\-wgb16a|xn\\-xkc2al3hye2a|xn\\-xkc2dl3a5ee0h|xn\\-yfro4i67o|xn\\-ygb2ammx|xn\\-zckzah|xxx)"
+ "|y[et]"
+ "|z[amw]))";
}

/**
 * Regular expression to match all IANA top-level domains.
 *
 * List accurate as of 2015/11/24. List taken from:

```

```

* http://data.iana.org/TLD/tlds-alpha-by-domain.txt
* This pattern is auto-generated by frameworks/ex/common/tools/make-iana-tld-pattern.

y
*
* @hide
*/
static final String IANA_TOP_LEVEL_DOMAINS =
"(?:" +
+ "(?:aaa|aarp|abb|abbott|abogado|academy|accenture|accountant|accountants|aco|activ
"
+ "|actor|ads|adult|aeg|aero|afl|agency|aig|airforce|airtel|allfinanz|alsace|amica|amsterda
"
+ "|android|apartments|app|apple|aquarelle|aramco|archi|army|arpa|arte|asia|associates"
+ "|attorney|auction|audio|auto|autos|axa|azure|a[cdefgilmoqrstuwxz])"
+ "|(?:band|bank|bar|barcelona|barclaycard|barclays|bargains|bauhaus|bayern|bbc|bbva"
+ "|bcn|beats|beer|bentley|berlin|best|bet|bharti|bible|bid|bike|bing|bingo|bio|biz|black"
+ "|blackfriday|bloomberg|blue|bms|bmw|bnl|bnpparibas|boats|bom|bond|boo|boots|bo
tique"
+ "|bradesco|bridgestone|broadway|broker|brother|brussels|budapest|build|builders|busi
ess"
+ "|buzz|bzh|b[abdefghijlmnorstuvwxyz])"
+ "|(?:cab|cafe|cal|camera|camp|cancerresearch|canon|capetown|capital|car|caravan|cards
"
+ "|care|career|careers|cars|cartier|casa|cash|casino|cat|catering|cba|cbn|ceb|center|ceo"
+ "|cern|cfa|cfd|chanel|channel|chat|cheap|chloe|christmas|chrome|church|cipriani|cisco"
+ "|citic|city|cityeats|claims|cleaning|click|clinic|clothing|cloud|club|clubmed|coach"
+ "|codes|coffee|college|cologne|com|commbank|community|company|computer|comsec
condos"
+ "|construction|consulting|contractors|cooking|cool|coop|corsica|country|coupons|cour
es"
+ "|credit|creditcard|creditunion|cricket|crown|crs|cruises|csc|cuisinella|cymru|cyou|c[acdf
hiklmnoruvwxz])"
+ "|(?:dabur|dad|dance|date|dating|datsun|day|dclk|deals|degree|delivery|dell|delta"
+ "|democrat|dental|dentist|desi|design|dev|diamonds|diet|digital|direct|directory|discou
t"
+ "|dnp|docs|dog|doha|domains|doosan|download|drive|durban|dvag|d[ejkmoz])"
+ "|(?:earth|eat|edu|education|email|emerck|energy|engineer|engineering|enterprises"
+ "|epson|equipment|erni|esq|estate|eurovision|eus|events|everbank|exchange|expert|ex
osed"
+ "|express|e[cegrstu])"
+ "|(?:fage|fail|fairwinds|faith|family|fan|fans|farm|fashion|feedback|ferrero|film"
+ "|final|finance|financial|firmdale|fish|fishing|fit|fitness|flights|florist|flowers|flsmidth"
+ "|fly|foo|football|forex|forsale|forum|foundation|frl|frogans|fund|furniture|futbol|fyi"
+ "|f[ijkmor])"
+ "|(?:gal|gallery|game|garden|gbiz|gdn|gea|gent|genting|ggee|gift|gifts|gives|giving"
+ "|glass|gle|global|globol|gmail|gmo|gmx|gold|goldpoint|golf|goo|goog|google|gop|gov
grainger"
+ "|graphics|gratis|green|gripe|group|gucci|guge|guide|guitars|guru|g[abdefghilmnpqrst
wy])"
+ "|(?:hamburg|hangout|haus|healthcare|help|here|hermes|hiphop|hitachi|hiv|hockey|hold
nings"
+ "|holiday|homedepot|homes|honda|horse|host|hosting|hoteles|hotmail|house|how|hsbc
hyundai"
+ "|h[kmnrtu])"

```

+ "|(?:ibm|icbc|ice|icu|ifm|iinet|immo|immobilien|industries|infiniti|info|ing|ink|institute"
 + "|insure|int|international|investments|ipiranga|irish|ist|istanbul|itau|iwc|i[delmnoqrst])"
 + "|(?:jaguar|javajcb|jetzt|jewelry|jlcjll|jobs|joburg|jprs|juegos|j[emop])"
 + "|(?:kaufen|kddi|kia|kim|kinder|kitchen|kiwi|koeln|komatsu|krd|kred|kyoto|k[eghimnprw
 z])"
 + "|(?:lacaixa|lancaster|land|landrover|lasalle|lat|latrobe|law|lawyer|lds|lease|leclerc"
 + "|legal|lexus|lgbt|liaison|idl|life|lifestyle|lighting|limited|limo|linde|link|live"
 + "|lixil|loan|loans|lol|london|lotte|lotto|love|ltd|ltda|lupin|luxe|luxury|l[abcikrstuvy])"
 + "|(?:madrid|maif|maison|man|management|mango|market|marketing|markets|marriott
 mba)"
 + "|media|meet|melbourne|meme|memorial|men|menu|meo|miami|microsoft|mil|mini|
 ma|mobi|moda"
 + "|moe|moi|mom|monash|money|montblanc|mormon|mortgage|moscow|motorcycles|
 ov|movie|movistar"
 + "|mtn|mtpc|mtr|museum|mutuelle|m[acdeghklmnopqrstuvwxyz])"
 + "|(?:nadex|nagoya|name|navy|nec|net|netbank|network|neustar|new|news|nexus|ngo|n
 k"
 + "|nico|ninja|nissan|nokia|nra|nrw|ntt|nyc|n[acefgilopruz])"
 + "|(?:obi|office|okinawa|omega|one|ong|onl|online|ooo|oracle|orange|org|organic|osaka|
 + "|otsuka|ovh|om)"
 + "|(?:page|panerai|paris|partners|parts|party|pet|pharmacy|philips|photo|photography|
 + "|photos|physio|piaget|pics|pictet|pictures|ping|pink|pizza|place|play|playstation|plumb
 ng"
 + "|plus|pohl|poker|porn|post|praxi|press|pro|prod|productions|prof|properties|property"
 + "|protection|pub|p[aefghklmnrtwy])"
 + "|(?:qpon|quebec|qa)"
 + "|(?:racing|realtor|realty|recipes|red|redstone|rehab|reise|reisen|reit|ren|rent|rentals|
 + "|repair|report|republican|rest|restaurant|review|reviews|rich|ricoh|rio|rip|rocher|rocks|
 + "|rodeo|rsvp|ruhr|run|rwe|ryukyu|r[eosuw])"
 + "|(?:saarland|sakura|sale|samsung|sandvik|sandvikcoromant|sanofi|sap|sapo|sarli|saxo|
 + "|sbs|sca|scb|schmidt|scholarships|school|schule|schwarz|science|scor|scot|seat|security|
 + "|seek|sener|services|seven|sew|sex|sexy|shiksha|shoes|show|shriram|singles|site|ski|
 + "|sky|skype|snclf|soccer|social|software|sohu|solar|solutions|sony|soy|space|spiegel|spre
 dbetting"
 + "|srl|stada|starhub|statoil|stc|stcgroup|stockholm|studio|study|style|sucks|supplies|
 + "|supply|support|surf|surgery|suzuki|swatch|swiss|sydney|systems|s[abcdehijklmnortu
 xyz])"
 + "|(?:tab|taipei|tatamotors|tatar|tattoo|taxi|taxi|team|tech|technology|tel|telefonica|
 + "|temasek|tennis|thd|theater|theatre|tickets|tienda|tips|tires|tiroll|today|tokyo|tools|
 + "|top|toray|toshiba|tours|town|toyota|toys|trade|trading|training|travel|trust|tui|t[cdfghj
 lmnotvwz])"
 + "|(?:ubs|university|uno|uol|u[agksyz])"
 + "|(?:vacations|vana|vegas|ventures|versicherung|vet|viajes|video|villas|vin|virgin|
 + "|vision|vista|vistaprint|viva|vlaanderen|vodka|vote|voting|voto|voyage|v[aceginu])"
 + "|(?:wales|walter|wang|watch|webcam|website|wed|wedding|weir|whoswho|wien|wiki|wi
 liamhill|"
 + "|win|windows|wine|wme|work|works|world|wtc|wtf|w[fs])"
 + "|(?:\u03b5|\u03bb|\u0431|\u0435|\u043b|\u0434|\u0435|\u0442|\u0438|\u043a|\u043e|\u
 43c|\u043c|\u043a|\u0434|"
 + "|\\u043c\\u043e\\u043d|\\u043c\\u043e\\u0441\\u043a\\u0432\\u0430|\\u043e\\u043d\\u043
 2\\u0441\\u0440\\u0431|"
 + "|\\u043e\\u0440\\u0433|\\u0440\\u0443\\u0441|\\u0440\\u0444|\\u0441\\u0430\\u0439\\u04
 2\\u0441\\u0440\\u0431"

+ "|\\u0443\\u043a\\u0440|\\u049b\\u0430\\u0437|\\u0570\\u0561\\u0575|\\u05e7\\u05d5\\u05d\\u0627\\u0631\\u0627\\u0645\\u0643\\u0648"
+ "|\\u0627\\u0644\\u0627\\u0631\\u062f\\u0646|\\u0627\\u0644\\u062c\\u0632\\u0627\\u0626\\u0631\\u0627\\u0644\\u0633\\u0639\\u0648\\u062f\\u064a\\u0629"
+ "|\\u0627\\u0644\\u0645\\u063a\\u0631\\u0628|\\u0627\\u0645\\u0627\\u0631\\u0627\\u062a\\u0627\\u06cc\\u0631\\u0627\\u0646"
+ "|\\u0628\\u0627\\u0632\\u0627\\u0631|\\u0628\\u06be\\u0627\\u0631\\u062a|\\u062a\\u064\\u0646\\u0633"
+ "|\\u0633\\u0648\\u062f\\u0627\\u0646|\\u0633\\u0648\\u0631\\u064a\\u0629|\\u0634\\u0628\\u0643\\u0629"
+ "|\\u0639\\u0631\\u0627\\u0642|\\u0639\\u0645\\u0627\\u0646|\\u0641\\u0644\\u0633\\u063\\u064a\\u0646"
+ "|\\u0642\\u0637\\u0631|\\u0643\\u0648\\u0645|\\u0645\\u0635\\u0631|\\u0645\\u0644\\u06a\\u0633\\u064a\\u0627"
+ "|\\u0645\\u0648\\u0642\\u0639|\\u0915\\u0949\\u092e|\\u0928\\u0947\\u091f|\\u092d\\u093\\u0930\\u0924"
+ "|\\u0938\\u0902\\u0917\\u0920\\u0928|\\u09ad\\u09be\\u09b0\\u09a4|\\u0a2d\\u0a3e\\u0a3\\u0a24|\\u0aad\\u0abe\\u0ab0\\u0aa4"
+ "|\\u0b87\\u0ba8\\u0bcd\\u0ba4\\u0bbf\\u0baf\\u0bbe|\\u0b87\\u0bb2\\u0b99\\u0bcd\\u0b9\\u0bc8\\u0b9a\\u0bbf\\u0b99\\u0bcd\\u0b95\\u0baa\\u0bcd\\u0baa\\u0bc2\\u0bb0\\u0bcd"
+ "|\\u0c2d\\u0c3e\\u0c30\\u0c24\\u0c4d|\\u0dbd\\u0d82\\u0d9a\\u0dcf|\\u0e04\\u0e2d\\u0e21\\u0e44\\u0e17\\u0e22"
+ "|\\u10d2\\u10d4|\\u307f\\u3093\\u306a|\\u30b0\\u30fc\\u30b0\\u30eb|\\u30b3\\u30e0\\u4e1\\u754c"
+ "|\\u4e2d\\u4fe1|\\u4e2d\\u56fd|\\u4e2d\\u570b|\\u4e2d\\u6587\\u7f51|\\u4f01\\u4e1a|\\u4f5\\u5c71"
+ "|\\u4fe1\\u606f|\\u5065\\u5eb7|\\u516b\\u5366|\\u516c\\u53f8|\\u516c\\u76ca|\\u53f0\\u6e7e\\u53f0\\u7063"
+ "|\\u5546\\u57ce|\\u5546\\u5e97|\\u5546\\u6807|\\u5728\\u7ebf|\\u5927\\u62ff|\\u5a31\\u4e50\\u5de5\\u884c"
+ "|\\u5e7f\\u4e1c|\\u6148\\u5584|\\u6211\\u7231\\u4f60|\\u624b\\u673a|\\u653f\\u52a1|\\u653f\\u5e9c"
+ "|\\u65b0\\u52a0\\u5761|\\u65b0\\u95fb|\\u65f6\\u5c1a|\\u673a\\u6784|\\u6de1\\u9a6c\\u9521\\u6e38\\u620f"
+ "|\\u70b9\\u770b|\\u79fb\\u52a8|\\u7ec4\\u7ec7\\u673a\\u6784|\\u7f51\\u5740|\\u7f51\\u5e97\\u7f51\\u7edc"
+ "|\\u8c37\\u6b4c|\\u96c6\\u56e2|\\u98de\\u5229\\u6d66\\u9910\\u5385\\u9999\\u6e2f\\ub27\\ub137"
+ "|\\ub2f7\\ucef4\\uc0bc\\uc131\\ud55c\\uad6d|xbox"
+ "|xerox|xin|xn\\-\\-11b4c3d|xn\\-\\-1qqw23a|xn\\-\\-30rr7y|xn\\-\\-3bst00m|xn\\-\\-3d443g"
+ "|xn\\-\\-3e0b707e|xn\\-\\-3pxu8k|xn\\-\\-42c2d9a|xn\\-\\-45brj9c|xn\\-\\-45q11c|xn\\-\\-4gbrim"
+ "|xn\\-\\-55qw42g|xn\\-\\-55qx5d|xn\\-\\-6frz82g|xn\\-\\-6qq986b3xl|xn\\-\\-80adxhk"
+ "|xn\\-\\-80ao21a|xn\\-\\-80asehdb|xn\\-\\-80aswg|xn\\-\\-90a3ac|xn\\-\\-90ais|xn\\-\\-9dbq2a"
+ "|xn\\-\\-9et52u|xn\\-\\-b4w605ferd|xn\\-\\-c1avg|xn\\-\\-c2br7g|xn\\-\\-cg4bki|xn\\-\\-clchc0ea0b2g2a9gcd"
+ "|xn\\-\\-czr694b|xn\\-\\-czrs0t|xn\\-\\-czru2d|xn\\-\\-d1acj3b|xn\\-\\-d1alf|xn\\-\\-efv88h"
+ "|xn\\-\\-estv75g|xn\\-\\-fhbei|xn\\-\\-fiq228c5hs|xn\\-\\-fiq64b|xn\\-\\-fiqs8s|xn\\-\\-fqz9s"
+ "|xn\\-\\-fjq720a|xn\\-\\-flw351e|xn\\-\\-fpcrj9c3d|xn\\-\\-fzc2c9e2c|xn\\-\\-gecrj9c"

```

+ "|xn\\-\\-h2brj9c|xn\\-\\-hxt814e|xn\\-\\-i1b6b1a6a2e|xn\\-\\-imr513n|xn\\-\\-io0a7i"
+ "|xn\\-\\-j1aef|xn\\-\\-j1amh|xn\\-\\-j6w193g|xn\\-\\-kcrx77d1x4a|xn\\-\\-kprw13d|xn\\
-\\-kqry57d"
+ "|xn\\-\\-kput3i|xn\\-\\-l1acc|xn\\-\\-lgbbat1ad8j|xn\\-\\-mgb9awbf|xn\\-\\-mgba3a3e
t"
+ "|xn\\-\\-mgba3a4f16a|xn\\-\\-mgbaam7a8h|xn\\-\\-mgbab2bd|xn\\-\\-mgbayh7gpa|
n\\-\\-mgbbh1a71e"
+ "|xn\\-\\-mgbc0a9azcg|xn\\-\\-mgberp4a5d4ar|xn\\-\\-mgbpl2fh|xn\\-\\-mgbtx2b|xn\\
-\\-mgbx4cd0ab"
+ "|xn\\-\\-mk1bu44c|xn\\-\\-mxtq1m|xn\\-\\-ngbc5azd|xn\\-\\-node|xn\\-\\-nqv7f|xn\\-
\\-nqv7fs00ema"
+ "|xn\\-\\-nyqy26a|xn\\-\\-o3cw4h|xn\\-\\-ogbpf8fl|xn\\-\\-p1acf|xn\\-\\-p1ai|xn\\-\\-p
bs0dh"
+ "|xn\\-\\-pssy2u|xn\\-\\-q9jyb4c|xn\\-\\-qcka1pmc|xn\\-\\-qxam|xn\\-\\-rhqv96g|xn\\-
\\-s9brj9c"
+ "|xn\\-\\-ses554g|xn\\-\\-t60b56a|xn\\-\\-tckwe|xn\\-\\-unup4y|xn\\-\\-vermgensbera
er\\-ctb"
+ "|xn\\-\\-vermgensberatung\\-pwb|xn\\-\\-vhquv|xn\\-\\-vuq861b|xn\\-\\-wgbh1c|xn\\
-\\-wgb16a"
+ "|xn\\-\\-xhq521b|xn\\-\\-xkc2al3hye2a|xn\\-\\-xkc2dl3a5ee0h|xn\\-\\-y9a3aq|xn\\-\\-
fro4i67o"
+ "|xn\\-\\-ygb12ammx|xn\\-\\-zfr164b|xperia|xxx|xyz)"
+ "|(?:yachts|yamaxun|yandex|yodobashi|yoga|yokohama|youtube|y[et])"
+ "|(?:zara|zip|zone|zuerich|z[amw]))";

```

/**
 * Kept for backward compatibility reasons.
 */
 * @deprecated Deprecated since it does not include all IRI characters defined in RFC 3987
 */
@Deprecated
public static final String GOOD_IRI_CHAR =
 "a-zA-Z0-9\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF";

public static final Pattern IP_ADDRESS
= Pattern.compile(
 "((25[0-5]|2[0-4][0-9]|0[1-9][0-9]{2})|[1-9][0-9]{2})\\.(25[0-5]|2[0-4]"
 + "[0-9]{2}|0[1-9][0-9]{2})|[1-9][0-9]{2}|0)\\.(25[0-5]|2[0-4][0-9]|0[1-9]"
 + "[0-9]{2})|[1-9][0-9]{2}|0)\\.(25[0-5]|2[0-4][0-9]|0[1-9]{2}"
 + "|[1-9][0-9]{2}))");

/**
 * Valid UCS characters defined in RFC 3987. Excludes space characters.
 */
private static final String UCS_CHAR = "[" +
 "\u00A0-\uD7FF" +
 "\uF900-\uFDCF" +
 "\uFDF0-\uFFEF" +
 "\uD800\uDC00-\uD83F\uDFFD" +
 "\uD840\uDC00-\uD87F\uDFFD" +
 "\uD880\uDC00-\uD8BF\uDFFD" +
 "\uD8C0\uDC00-\uD8FF\uDFFD" +
 "\uD900\uDC00-\uD93F\uDFFD" +
 "\uD940\uDC00-\uD97F\uDFFD" +


```

/**
 * Regular expression pattern to match most part of RFC 3987
 * Internationalized URLs, aka IRIs.
 */
public static final Pattern WEB_URL = Pattern.compile(
    + "("
    + "(?:" + PROTOCOL + "(?:" + USER_INFO + "?")?" + ")"
    + "(?:" + DOMAIN_NAME + ")"
    + "(?:" + PORT_NUMBER + "?"
    + ")"
    + "(" + PATH_AND_QUERY + "?"
    + WORD_BOUNDARY
    + ")");
}

/**
 * Regular expression that matches known TLDs and punycode TLDs
 */
private static final String STRICT_TLD = "(?:" +
    IANA_TOP_LEVEL_DOMAINS + "|" + PUNYCODE_TLD + ")";

/**
 * Regular expression that matches host names using {@link #STRICT_TLD}
 */
private static final String STRICT_HOST_NAME = "(?:(?:" + IRI_LABEL + "\\.)+"
    + STRICT_TLD + ")";

/**
 * Regular expression that matches domain names using either {@link #STRICT_HOST_NAME}
E} or
 * {@link #IP_ADDRESS}
 */
private static final Pattern STRICT_DOMAIN_NAME =
    Pattern.compile("(?:" + STRICT_HOST_NAME + "|" + IP_ADDRESS + ")");

/**
 * Regular expression that matches domain names without a TLD
 */
private static final String RELAXED_DOMAIN_NAME =
    "(?:" + "(?:" + IRI_LABEL + "(?:\\.(?=\\$))" + "?)+"
    + "|"
    + IP_ADDRESS + ")";

/**
 * Regular expression to match strings that do not start with a supported protocol. The TLD
 * are expected to be one of the known TLDs.
 */
private static final String WEB_URL_WITHOUT_PROTOCOL = (
    + WORD_BOUNDARY
    + "(?<!:\\\\\\\\V)"
    + "("
    + "(?:" + STRICT_DOMAIN_NAME + ")"
    + "(?:" + PORT_NUMBER + "?"
    + ")"
    + "(" + PATH_AND_QUERY + "?"
    + WORD_BOUNDARY

```

```

        + ")";
    }

    /**
     * Regular expression to match strings that start with a supported protocol. Rules for domain names and TLDs are more relaxed. TLDs are optional.
     */
    private static final String WEB_URL_WITH_PROTOCOL = "("
        + WORD_BOUNDARY
        + "(?:" +
        + "(?:" + PROTOCOL + "(?:" + USER_INFO + ")?" + ")" +
        + "(?:" + RELAXED_DOMAIN_NAME + ")?" +
        + "(?:" + PORT_NUMBER + ")?" +
        + ")" +
        + "(?:" + PATH_AND_QUERY + ")?" +
        + WORD_BOUNDARY
        + ")";
}

/**
 * Regular expression pattern to match IIRIs. If a string starts with http(s)// the expression tries to match the URL structure with a relaxed rule for TLDs. If the string does not start with http(s)// the TLDs are expected to be one of the known TLDs.
 *
 * @hide
 */
public static final Pattern AUTOLINK_WEB_URL = Pattern.compile(
    "(" + WEB_URL_WITH_PROTOCOL + "|" + WEB_URL_WITHOUT_PROTOCOL + ")");

/**
 * Regular expression for valid email characters. Does not include some of the valid characters defined in RFC5321: #&~!^`{}=/=$*?|
 */
private static final String EMAIL_CHAR = LABEL_CHAR + "\\+\\-_%";

/**
 * Regular expression for local part of an email address. RFC5321 section 4.5.3.1.1 limits the local part to be at most 64 octets.
 */
private static final String EMAIL_ADDRESS_LOCAL_PART =
    "[" + EMAIL_CHAR + "]" + "(?:" + EMAIL_CHAR + "\\.{1,62}" + EMAIL_CHAR + ")]?";

/**
 * Regular expression for the domain part of an email address. RFC5321 section 4.5.3.1.2 limits the domain to be at most 255 octets.
 */
private static final String EMAIL_ADDRESS_DOMAIN =
    "(?=.{1,255}(?:\\s|\\$|^))" + HOST_NAME;

/**
 * Regular expression pattern to match email addresses. It excludes double quoted local parts and the special characters #&~!^`{}=/=$*?| that are included in RFC5321.
 */

```

```

* @hide
*/
public static final Pattern AUTOLINK_EMAIL_ADDRESS = Pattern.compile("(" + WORD_BOU
DARY +
    "(?:" + EMAIL_ADDRESS_LOCAL_PART + "@" + EMAIL_ADDRESS_DOMAIN + ")" +
    WORD_BOUNDARY + ")"
);

public static final Pattern EMAIL_ADDRESS
= Pattern.compile(
    "[a-zA-Z0-9\\+\\.\\_\\%\\-\\\\+]{1,256}" +
    "\\@"
    "[a-zA-Z0-9][a-zA-Z0-9\\-]{0,64}" +
    "(" +
        "\\." +
        "[a-zA-Z0-9][a-zA-Z0-9\\-]{0,25}" +
    ")"+
);

/** 
 * This pattern is intended for searching for things that look like they
 * might be phone numbers in arbitrary text, not for validating whether
 * something is in fact a phone number. It will miss many things that
 * are legitimate phone numbers.
 *
 * <p> The pattern matches the following:
 * <ul>
 * <li> Optionally, a + sign followed immediately by one or more digits. Spaces, dots, or da
hes
 * may follow.
 * <li> Optionally, sets of digits in parentheses, separated by spaces, dots, or dashes.
 * <li> A string starting and ending with a digit, containing digits, spaces, dots, and/or dash
s.
 * </ul>
 */
public static final Pattern PHONE
= Pattern.compile(          // sdd = space, dot, or dash
    "(\\+[0-9]+[\\-\\.]*?)"      // +<digits><sdd>*
    + "((\\([0-9]+\\)[\\-\\.]*?)" // (<digits>)<sdd>*
    + "([0-9][0-9\\-\\.]+[0-9])"; // <digit><digit|sdd>+<digit>

/** 
 * Convenience method to take all of the non-null matching groups in a
 * regex Matcher and return them as a concatenated string.
 *
 * @param matcher The Matcher object from which grouped text will
 *                 be extracted
 *
 * @return A String comprising all of the non-null matched
 *         groups concatenated together
 */
public static final String concatGroups(Matcher matcher) {
    StringBuilder b = new StringBuilder();
    final int numGroups = matcher.groupCount();

```

```
for (int i = 1; i <= numGroups; i++) {
    String s = matcher.group(i);

    if (s != null) {
        b.append(s);
    }
}

return b.toString();
}

/**
 * Convenience method to return only the digits and plus signs
 * in the matching string.
 *
 * @param matcher The Matcher object from which digits and plus will
 *                be extracted
 *
 * @return A String comprising all of the digits and plus in
 *         the match
 */
public static final String digitsAndPlusOnly(Matcher matcher) {
    StringBuilder buffer = new StringBuilder();
    String matchingRegion = matcher.group();

    for (int i = 0, size = matchingRegion.length(); i < size; i++) {
        char character = matchingRegion.charAt(i);

        if (character == '+' || Character.isDigit(character)) {
            buffer.append(character);
        }
    }
    return buffer.toString();
}

/**
 * Do not create this static utility class.
 */
private Patterns() {}
}
```