

How do you detect Credit card type based on number?

Asked 16 years, 3 months ago Modified 5 months ago

Viewed 574k times



601

I'm trying to figure out how to detect the type of credit card based purely on its number. Does anyone know of a definitive, reliable way to find this?



algorithm

language-agnostic

e-commerce

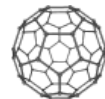


Share

Improve this question

Follow

edited Dec 30, 2011 at 16:15



NullUserException

85.4k ● 30 ● 211 ● 237

asked Sep 16, 2008 at 14:16



Andrew Edvalson

7,878 ● 5 ● 28 ● 24

-
- 4 Using a regular expression. Check out [this link](#) for more information. – [senfo](#) Sep 16, 2008 at 14:18
-
- 3 The details are all on Wikipedia: en.wikipedia.org/wiki/Credit_card_numbers – [Sten Vesterli](#) Sep 16, 2008 at 14:20
-
- 1 There's a good summary table in Wikipedia, at en.wikipedia.org/wiki/Credit_card_numbers. It's the first one

to six digits that tell the type and issuer of the card. – [Alex](#)
Sep 16, 2008 at 14:20

-
- 3 I wouldn't use a regex other than to pull out the first numeric group, you can generally tell just from the first 4 numbers (in the US). Also before bothering to pay for clearing a charge run a Mod 10 checksum on the card number to make sure it could be legitimate. [Luhn algorithm](#) – [Dan Blair](#) Sep 16, 2008 at 14:25
-
- 3 also can anyone comment if these algorithms are good 'for all time' - or do they periodically change, like for instance the algorithm for 'calculating if a phone number is in california' – [Simon_Weaver](#) Nov 22, 2009 at 23:20
-

29 Answers

Sorted by:

Highest score (default)



897



+50



The credit/debit card number is referred to as a **PAN**, or *Primary Account Number*. The first six digits of the PAN are taken from the **IIN**, or *Issuer Identification Number*, belonging to the issuing bank (IINs were previously known as BIN — Bank Identification Numbers — so you may see references to that terminology in some documents). These six digits are subject to an international standard, [ISO/IEC 7812](#), and can be used to determine the type of card from the number.

Unfortunately the actual ISO/IEC 7812 database is not publicly available, however, there are unofficial lists, both commercial and free, including [on Wikipedia](#).

Anyway, to detect the type from the number, you can use a regular expression like the ones below: [Credit for](#)

[original expressions](#)

Visa: `^4[0-9]{6,}$` Visa card numbers start with a 4.

MasterCard: `^5[1-5][0-9]{5,}|222[1-9][0-9]{3,}|22[3-9][0-9]{4,}|2[3-6][0-9]{5,}|27[01][0-9]{4,}|2720[0-9]{3,}$` Before 2016, MasterCard numbers start with the numbers 51 through 55, **but this will only detect MasterCard credit cards**; there are other cards issued using the MasterCard system that do not fall into this IIN range. In 2016, they will add numbers in the range (222100-272099).

American Express: `^3[47][0-9]{5,}$` American Express card numbers start with 34 or 37.

Diners Club: `^3(?:0[0-5]|[68][0-9])[0-9]{4,}$` Diners Club card numbers begin with 300 through 305, 36 or 38. There are Diners Club cards that begin with 5 and have 16 digits. These are a joint venture between Diners Club and MasterCard and should be processed like a MasterCard.

Discover: `^6(?:011|5[0-9]{2})[0-9]{3,}$` Discover card numbers begin with 6011 or 65.

JCB: `^(?:2131|1800|35[0-9]{3})[0-9]{3,}$` JCB cards begin with 2131, 1800 or 35.

Unfortunately, there are a number of card types processed with the MasterCard system that do not live in MasterCard's IIN range (numbers starting 51...55); the

most important case is that of Maestro cards, many of which have been issued from other banks' IIN ranges and so are located all over the number space. As a result, **it may be best to assume that any card that is not of some other type you accept must be a MasterCard.**

Important: card numbers do vary in length; for instance, Visa has in the past issued cards with 13 digit PANs and cards with 16 digit PANs. Visa's documentation currently indicates that it may issue or may have issued numbers with between 12 and 19 digits. **Therefore, you should not check the length of the card number, other than to verify that it has at least 7 digits** (for a complete IIN plus one check digit, which should match the value predicted by [the Luhn algorithm](#)).

One further hint: **before processing a cardholder PAN, strip any whitespace and punctuation characters from the input.** Why? Because it's typically *much* easier to enter the digits in groups, similar to how they're displayed on the front of an actual credit card, i.e.

4444 4444 4444 4444

is much easier to enter correctly than

444444444444444444

There's really no benefit in chastising the user because they've entered characters you don't expect here.

This also implies making sure that your entry fields have room for *at least* 24 characters, otherwise users who enter spaces will run out of room. I'd recommend that you make the field wide enough to display 32 characters and allow up to 64; that gives plenty of headroom for expansion.

Here's an image that gives a little more insight:

UPDATE (2016): Mastercard is to implement new BIN ranges starting [Ach Payment](#).



CRACKING THE CREDIT CARD CODE

Big Bank

4417 1234 5678 9113

VALID 8/11

LEE M CARDHOLDER

VISTA

The first digit is the **Major Industry Identifier**. It designates the category of the entity which issued the card.

1 and 2 are Airlines,
3 is Travel and Entertainment
4 and 5 are Banking and Financial
6 is Merchandizing and Banking
7 is Petroleum
8 is Telecommunications
9 is National assignment

The 7th and following digits, excluding the final digit, are the person's account number. This leaves a trillion possible combinations if the maximum of 12 digits is used. Many cards only use 9 digits.


The final digit is the check digit or checksum. It is used to validate the credit card number using the Luhn algorithm.

The first 6 digits are the **Issuer Identification Number**. It will identify the institution that issued the card.

Visa: 4xxxxx
Mastercard 51xxxx - 55xxxx
Discover: 6011xx, 644xxx, 65xxxx
Amex: 34xxxx, 37xxxx

Cards can be looked up by their **IIN**. A card that starts with 376211 is a Singapore Airlines Krisflyer American Express Gold Card.

How to Validate a Credit Card With



529962 designates a pre-paid Much-Music MasterCard.

Your Mind

Take the above number (or any credit card number)

4417 1234 5678 9113

And double every other digit from the right.

4417 1234 5678 9113
 $\times 2 \quad \times 2 \quad \times 2 \quad \times 2 \quad \times 2 \quad \times 2 \quad \times 2 \quad \times 2$

8 2 2 6 10 14 18 2

Add these new digits to undoubled ones.

4 7 2 4 6 8 1 3

All double digit numbers are added as a sum of their digits, so 14 becomes 1+4.

$$8+4+2+7 + 2+2+6+4 + 1+0+6+1+4+8 + 1+8+1+2+3 \\ = 70$$

If the final sum is divisible by 10, then the credit card number is valid.
If it's not divisible by 10, the number is invalid or fake. Try it and see.



byJess.Net
mint.com/blog

Share Improve this answer

edited Jun 21, 2021 at 16:03

Follow



ArtOfWarfare

21.4k ● 19 ● 149 ● 199

answered Sep 16, 2008 at 14:18




senfo


29k ● 16 ● 78 ● 106

8 great example. do you have the regular expression for maestro cards? – [Manikandan](#) May 6, 2013 at 10:26

8 NO, no, no. You cannot rely on the lengths of card numbers; they can change at any time. The only part of the card number you can rely on is the IIN (which used to be called a BIN) and which is a prefix of the number. Additionally, you *cannot* detect Mastercard cards in the manner you suggest; that will only pick up a subset of the cards that are processed via the Mastercard system (the main problem being Maestro cards, which have a variety of IIN prefixes). – [al45tair](#) Feb 4, 2014 at 8:03

4 @senfo You're right, 5412 would not be a complete Mastercard number. IINs are six digits long, so a complete card number must be 7 digits (minimum) and must pass the Luhn check. There's no need for "proof" that Mastercard numbers have anything other than 16 digits; the point is that, regardless of the situation today, in future they might issue cards with 17 or 18 digits, or for that matter some with 15. Relying on them being 16 digits long is unnecessary and creates a long-term maintenance risk. – [al45tair](#) Feb 5, 2014 at 7:51 

4 I find it very hard to believe that some valid cards would not have a correct check digit according to the Luhn algorithm. It is used absolutely everywhere to check card numbers against simple typos and dumb fraud attempts. Instead, I have observed some quite smart people simply not grasp the algorithm, and they just calculate it wrong. – [Rennex](#) Jun 8, 2015 at 18:02

4 @BaileyParker—the [LUHN algorithm](#) doesn't require the number to be divisible by 10 (or any particular number), it simply applies a formula to generate a value from the digits then looks at the last digit of the value (it uses %10, not /10). It's used by [all cards in use](#). – [RobG](#) Mar 8, 2016 at 6:29 



In javascript:

97



```
function detectCardType(number) {
  var re = {
    electron: /^(4026|417500|4405|4508|4844|4913|4
    maestro:
    /^(5018|5020|5038|5612|5893|6304|6759|6761|6762|6763|0
    dankort: /^(5019)\d+$/,
    interpayment: /^(636)\d+$/,
    unionpay: /^(62|88)\d+$/,
    visa: /^[0-9]{12}(?:[0-9]{3})?$/,
    mastercard: /^5[1-5][0-9]{14}$/ ,
    amex: /^3[47][0-9]{13}$/ ,
    diners: /^3(?:0[0-5]|[68][0-9])[0-9]{11}$/ ,
    discover: /^6(?:011|5[0-9]{2})[0-9]{12}$/ ,
    jcb: /^(?:2131|1800|35\d{3})\d{11}$/
  }

  for(var key in re) {
    if(re[key].test(number)) {
      return key
    }
  }
}
```

Unit test:

```
describe('CreditCard', function() {
  describe('#detectCardType', function() {

    var cards = {
      '8800000000000000': 'UNIONPAY',

      '4026000000000000': 'ELECTRON',
      '4175000000000000': 'ELECTRON',
      '4405000000000000': 'ELECTRON',
      '4508000000000000': 'ELECTRON',
      '4844000000000000': 'ELECTRON',
      '4913000000000000': 'ELECTRON',
```

'4917000000000000' : 'ELECTRON',
'5019000000000000' : 'DANKORT',

'5018000000000000' : 'MAESTRO',
'5020000000000000' : 'MAESTRO',
'5038000000000000' : 'MAESTRO',
'5612000000000000' : 'MAESTRO',
'5893000000000000' : 'MAESTRO',
'6304000000000000' : 'MAESTRO',
'6759000000000000' : 'MAESTRO',
'6761000000000000' : 'MAESTRO',
'6762000000000000' : 'MAESTRO',
'6763000000000000' : 'MAESTRO',
'0604000000000000' : 'MAESTRO',
'6390000000000000' : 'MAESTRO',

'3528000000000000' : 'JCB',
'3589000000000000' : 'JCB',
'3529000000000000' : 'JCB',

'6360000000000000' : 'INTERPAYMENT',

'4916338506082832' : 'VISA',
'4556015886206505' : 'VISA',
'4539048040151731' : 'VISA',
'4024007198964305' : 'VISA',
'4716175187624512' : 'VISA',

'5280934283171080' : 'MASTERCARD',
'5456060454627409' : 'MASTERCARD',
'5331113404316994' : 'MASTERCARD',
'5259474113320034' : 'MASTERCARD',
'5442179619690834' : 'MASTERCARD',

'6011894492395579' : 'DISCOVER',
'6011388644154687' : 'DISCOVER',
'6011880085013612' : 'DISCOVER',
'6011652795433988' : 'DISCOVER',
'6011375973328347' : 'DISCOVER',

'345936346788903' : 'AMEX',
'377669501013152' : 'AMEX',
'373083634595479' : 'AMEX',

```

        '370710819865268': 'AMEX',
        '371095063560404': 'AMEX'
    };

    Object.keys(cards).forEach(function(number) {
        it('should detect card ' + number + ' as '
function() {
            Basket.detectCardType(number).should.e
        });
    });
});
});
});

```

Share Improve this answer

Follow

edited Jul 20, 2016 at 14:16



Martijn Scheffer

691 ● 7 ● 9

answered Oct 2, 2013 at 14:04



Anatoliy

30.1k ● 5 ● 47 ● 47

3 @jolly.exe - Your fiddle returns undefined for all tests. Doesn't work :(– [ShadeTreeDeveloper](#) Jul 22, 2015 at 12:38

1 @ShadeTreeDeveloper just enter any value eg. 372176090165471 for AMAX in text field – [Code Spy](#) Jul 22, 2015 at 13:45

@jolly.exe I see... I was hoping for something that would format as I type (off the keyup event). The fiddle does work when I enter a full number. – [ShadeTreeDeveloper](#) Jul 25, 2015 at 12:38

I ended up writing this bit of code to do the input formatting and validation that I wanted. quercusv.github.io/smartForm – [ShadeTreeDeveloper](#) Jul 30, 2015 at 21:31



Updated: 15th June 2016 (as an ultimate solution currently)

57



Please note that I even give vote up for the one is top voted, but to make it clear these are the regexps actually works i tested it with thousands of real BIN codes. **The most important is to use start strings (^) otherwise it will give false results in real world!**



JCB `^(?:2131|1800|35)[0-9]{0,}$` Start with: **2131, 1800, 35 (3528-3589)**

American Express `^3[47][0-9]{0,}$` Start with: **34, 37**

Diners Club `^3(?:0[0-59]{1}|[689])[0-9]{0,}$` Start with: **300-305, 309, 36, 38-39**

Visa `^4[0-9]{0,}$` Start with: **4**

MasterCard `^(5[1-5]|222[1-9]|22[3-9]|2[3-6]|27[01]|2720)[0-9]{0,}$` Start with: **2221-2720, 51-55**

Maestro `^(5[06789]|6)[0-9]{0,}$` Maestro always growing in the range: **60-69**, started with / not something else, but starting 5 must be encoded as mastercard anyway. Maestro cards must be detected in the end of the code because some others has in the range of 60-69. Please look at the code.

Discover `^(6011|65|64[4-9]|62212[6-9]|6221[3-9]|622[2-8]|6229[01]|62292[0-5])[0-9]{0,}$` Discover quite difficult to code, start with: **6011, 622126-622925, 644-649, 65**

In **javascript** I use this function. This is good when u assign it to an onkeyup event and it give result as soon as possible.

```
function cc_brand_id(cur_val) {
    // the regular expressions check for possible matches
    // the OR operators based on the number of chars
    // regexp string length {0} provided for soonest detection
    // the card numbers this way it could be used for BIN COD

    //JCB
    jcb_regex = new RegExp('^(:2131|1800|35)[0-9]{0,}(3528-3589)');
    // American Express
    amex_regex = new RegExp('^3[47][0-9]{0,}$'); //34,
    // Diners Club
    diners_regex = new RegExp('^3(?:0[0-59]{1}|[689])[0-9]{0,}(309, 36, 38-39)');
    // Visa
    visa_regex = new RegExp('^4[0-9]{0,}$'); //4
    // MasterCard
    mastercard_regex = new RegExp('^((5[1-5]|222[1-9]|26[0-9]|27[01]|2720)[0-9]{0,}$)'); //2221-2720, 51-55
    maestro_regex = new RegExp('^((5[06789]|6)[0-9]{0,}$)'); //60-69, started with / not something else, but encoded as mastercard anyway
    //Discover
    discover_regex = new RegExp('^((6011|65|64[4-9]|622[2-8]|6229[01]|62292[0-5])[0-9]{0,}$)');
    //6011, 622126-622925, 644-649, 65

    // get rid of anything but numbers
    cur_val = cur_val.replace(/\D/g, '');
}
```



```

        // checks per each, as their could be multiple hit
        //fix: ordering matter in detection, otherwise can
        rare cases
        var sel_brand = "unknown";
        if (cur_val.match(jcb_regex)) {
            sel_brand = "jcb";
        } else if (cur_val.match(amex_regex)) {
            sel_brand = "amex";
        } else if (cur_val.match(diners_regex)) {
            sel_brand = "diners_club";
        } else if (cur_val.match(visa_regex)) {
            sel_brand = "visa";
        } else if (cur_val.match(mastercard_regex)) {
            sel_brand = "mastercard";
        } else if (cur_val.match(discover_regex)) {
            sel_brand = "discover";
        } else if (cur_val.match(maestro_regex)) {
            if (cur_val[0] == '5') { //started 5 must be m
                sel_brand = "mastercard";
            } else {
                sel_brand = "maestro"; //maestro is all 60
                something else, thats why this condition in the end
            }
        }

        return sel_brand;
    }
}

```

Here you can play with it:

<http://jsfiddle.net/upN3L/69/>

**For PHP use this function, this detects some sub
VISA/MC cards too:**

```

/**
 * Obtain a brand constant from a PAN
 *
 * @param string $pan          Credit card number
 * @param bool   $include_sub_types Include detection
 * @return string

```

```

*/
public static function getCardBrand($pan, $include_sub
{
    //maximum length is not fixed now, there are growi
    numbers in length, limiting can give false negatives a

    //these regexps accept not whole cc numbers too
    //visa
    $visa_regex = "/^4[0-9]{0,}$"/;
    $vpreca_regex = "/^428485[0-9]{0,}$"/;
    $postepay_regex = "/^(402360|402361|403035|417631|
    $cartasi_regex = "/^(432917|432930|453998)[0-9]{0,
    $entropay_regex = "/^(406742|410162|431380|459061|
    {0,}$"/;
    $o2money_regex = "/^(422793|475743)[0-9]{0,}$"/;

    // MasterCard
    $mastercard_regex = "/^(5[1-5]|222[1-9]|22[3-9]|2[
    {0,}$"/;
    $maestro_regex = "/^(5[06789]|6)[0-9]{0,}$"/;
    $kukuruza_regex = "/^525477[0-9]{0,}$"/;
    $yunacard_regex = "/^541275[0-9]{0,}$"/;

    // American Express
    $amex_regex = "/^3[47][0-9]{0,}$"/;

    // Diners Club
    $diners_regex = "/^3(?:0[0-59]{1}|[689])[0-9]{0,}$

    //Discover
    $discover_regex = "/^(6011|65|64[4-9]|62212[6-9]|6
    8]|6229[01]|62292[0-5])[0-9]{0,}$"/;

    //JCB
    $jcb_regex = "/^(?:2131|1800|35)[0-9]{0,}$"/;

    //ordering matter in detection, otherwise can give
    cases
    if (preg_match($jcb_regex, $pan)) {
        return "jcb";
    }

    if (preg_match($amex_regex, $pan)) {
        return "amex";
    }
}

```

```
}

if (preg_match($diners_regex, $span)) {
    return "diners_club";
}

//sub visa/mastercard cards
if ($include_sub_types) {
    if (preg_match($vpreca_regex, $span)) {
        return "v-preca";
    }
    if (preg_match($postepay_regex, $span)) {
        return "postepay";
    }
    if (preg_match($cartasi_regex, $span)) {
        return "cartasi";
    }
    if (preg_match($entropay_regex, $span)) {
        return "entropay";
    }
    if (preg_match($o2money_regex, $span)) {
        return "o2money";
    }
    if (preg_match($kukuruza_regex, $span)) {
        return "kukuruza";
    }
    if (preg_match($yunacard_regex, $span)) {
        return "yunacard";
    }
}

if (preg_match($visa_regex, $span)) {
    return "visa";
}

if (preg_match($mastercard_regex, $span)) {
    return "mastercard";
}

if (preg_match($discover_regex, $span)) {
    return "discover";
}

if (preg_match($maestro_regex, $span)) {
```

```
        if ($span[0] == '5') { //started 5 must be mast
            return "mastercard";
        }
        return "maestro"; //maestro is all 60-69 which
        thats why this condition in the end

    }

    return "unknown"; //unknown for this system
}
```

Share Improve this answer

Follow

edited Jan 7, 2020 at 15:23



William Desportes

1,698 ● 3 ● 26 ● 37

answered Feb 7, 2014 at 1:21



Janos Szabo

1,352 ● 1 ● 15 ● 14

-
- 1 And please note, that this is only CC number detection and not validation. That is separated, should be a Luhn check...
– [Janos Szabo](#) Jun 15, 2016 at 16:15

Where is Visa Electron, and why does the Maestro check return MasterCard in some cases? Shouldn't the MasterCard check that itself? – [BadHorsie](#) Aug 8, 2016 at 17:39

It fails to recognize this JCB test number as any of the types (3088514174175777), and identifies this test JCB number as diners_club (3096278649822922). Assuming this list of test card numbers are valid anyway (freeformatter.com/credit-card-number-generator-validator.html) – [Drew](#) Oct 4, 2016 at 18:56

there are no documentation that starting 308 or 309 could be a JCB card – [Janos Szabo](#) Oct 5, 2016 at 6:23

+1 for providing cc type detection code, which is what you typically want to do for the ux - the regex for the new range

on MC needs a small tweek: `/^(5[1-5]|222[1-9]|22[3-9][0-9]|2[3-6][0-9]{2}|27[01][0-9]|2720)[0-9]{0,}$/` – [kinakuta](#) Dec 29, 2016 at 23:45



26



```
public string GetCreditCardType(string CreditCardNumbe
{
    Regex regVisa = new Regex("^4[0-9]{12}(?:[0-9]{3})
    Regex regMaster = new Regex("^5[1-5][0-9]{14}$");
    Regex regExpress = new Regex("^3[47][0-9]{13}$");
    Regex regDiners = new Regex("^3(?:0[0-5]| [68][0-9]
    Regex regDiscover = new Regex("^6(?:011|5[0-9]{2})
    Regex regJCB = new Regex("^(?:2131|1800|35\\d{3})\\

    if (regVisa.IsMatch(CreditCardNumber))
        return "VISA";
    else if (regMaster.IsMatch(CreditCardNumber))
        return "MASTER";
    else if (regExpress.IsMatch(CreditCardNumber))
        return "AEXPRESS";
    else if (regDiners.IsMatch(CreditCardNumber))
        return "DINERS";
    else if (regDiscover.IsMatch(CreditCardNumber))
        return "DISCOVERS";
    else if (regJCB.IsMatch(CreditCardNumber))
        return "JCB";
    else
        return "invalid";
}
```

Here is the function to check Credit card type using
Regex , c#

Share Improve this answer

Follow

edited Jan 8, 2020 at 13:56



[William Desportes](#)

1,698 ● 3 ● 26 ● 37

answered Dec 12, 2012 at 14:55



Usman Younas

1,379 ● 15 ● 21



21



Check this out:

<http://www.breakingpar.com/bkp/home.nsf/0/87256B280015193F87256CC70060A01B>

```
function isValidCreditCard(type, ccnum) {
    /* Visa: length 16, prefix 4, dashes optional.
    Mastercard: length 16, prefix 51-55, dashes option
    Discover: length 16, prefix 6011, dashes optional.
    American Express: length 15, prefix 34 or 37.
    Diners: length 14, prefix 30, 36, or 38. */

    var re = new Regex({
        "visa": "/^4\d{3}-?\d{4}-?\d{4}-?\d",
        "mc": "/^5[1-5]\d{2}-?\d{4}-?\d{4}-?\d{4}$/",
        "disc": "/^6011-?\d{4}-?\d{4}-?\d{4}$/",
        "amex": "/^3[47]\d{13}$/",
        "diners": "/^3[068]\d{12}$/",
    })[type.toLowerCase()]

    if (!re.test(ccnum)) return false;
    // Remove all dashes for the checksum checks to el
    ccnum = ccnum.split("-").join("");
    // Checksum ("Mod 10")
    // Add even digits in even length strings or odd d
    strings.
    var checksum = 0;
    for (var i = (2 - (ccnum.length % 2)); i <= ccnum.
        checksum += parseInt(ccnum.charAt(i - 1));
    }
    // Analyze odd digits in even length strings or ev
    strings.
    for (var i = (ccnum.length % 2) + 1; i < ccnum.len
        var digit = parseInt(ccnum.charAt(i - 1)) * 2;
        if (digit < 10) { checksum += digit; } else {
    }
```

```
}  
if ((checksum % 10) == 0) return true;  
else return false;  
}
```

Share Improve this answer

edited Jan 8, 2020 at 14:42

Follow



William Desportes

1,698 ● 3 ● 26 ● 37

answered Mar 9, 2010 at 11:21



Rashy

911 ● 9 ● 14

Mastercard has upgraded and they now use numbers that start with 2[...] and so on. Please update your code. You may like to use this `^(?:5[1-5]|5[1-5][0-9]{14}|2(22[1-9][0-9]{12}|2[3-9][0-9]{13}|[3-6][0-9]{14}|7[0-1][0-9]{13}|720[0-9]{12}))$/` – Nicholas Mberev Jan 2, 2022 at 9:51



18

recently I needed such functionality, I was porting Zend Framework Credit Card Validator to ruby. ruby gem:

https://github.com/Fivell/credit_card_validations zend framework:



<https://github.com/zendframework/zf2/blob/master/library/Zend/Validator/CreditCard.php>



They both use INN ranges for detecting type. Here you can read [about INN](#)

According to this you can detect credit card alternatively (without regexps, but declaring some rules about prefixes and possible length)

So we have next rules for most used cards

```
##### most used brands #####

visa: [
    {length: [13, 16], prefixes: ['4']}
],
mastercard: [
    {length: [16], prefixes: ['51', '52',
'53', '54', '55']}
],

amex: [
    {length: [15], prefixes: ['34', '37']}
],
##### other brands #####
diners: [
    {length: [14], prefixes: ['300', '301',
'302', '303', '304', '305', '36', '38']},
],

#There are Diners Club (North America) cards
that begin with 5. These are a joint venture
between Diners Club and MasterCard, and are
processed like a MasterCard
# will be removed in next major version

diners_us: [
    {length: [16], prefixes: ['54', '55']}
],

discover: [
    {length: [16], prefixes: ['6011', '644',
'645', '646', '647', '648',
                                '649', '65']}
],

jcb: [
    {length: [16], prefixes: ['3528', '3529',
'353', '354', '355', '356', '357', '358', '1800']},
],
```

Then by searching prefix and comparing length you can detect credit card brand. Also don't forget about luhn algorithm (it is descibed here <http://en.wikipedia.org/wiki/Luhn>).

UPDATE

updated list of rules can be found here

https://raw.githubusercontent.com/Fivell/credit_card_validations/master/lib/data/brands.yaml

Share Improve this answer

edited Aug 9, 2019 at 10:00

Follow

answered Mar 19, 2013 at 12:15



Fivell

11.9k ● 3 ● 62 ● 101

2 Very illustrative. VISA cards may be 13 digits long.

– [Herman Kan](#) Jun 11, 2014 at 6:02

@HermanKan, no VISA website says it should be 16 length, I think long time ago it could be 13, but not nowadays – [Fivell](#) Jul 7, 2014 at 21:28

1 I think it is legacy support – [Fivell](#) Jul 8, 2014 at 10:11

1 @HermanKan, there is one more thing, VISA has VPay cards and accroding to wikipedia Visa's VPay brand can specify PAN lengths from 13 to 19 digits an so card number of more than 16 digits are now being seen. – [Fivell](#) Jul 8, 2014 at 10:14

1 @Ethan, check last link in my updated answer raw.githubusercontent.com/Fivell/credit_card_validations/mas



Here's [Complete C# or VB code for all kinds of CC related things](#) on codeproject.

13



- IsValidNumber
- GetCardTypeFromNumber
- GetCardTestNumber
- PassesLuhnTest



This article has been up for a couple years with no negative comments.

Share Improve this answer

Follow

edited Mar 12, 2014 at 20:44



[RoLYroLLs](#)

3,205 ● 5 ● 41 ● 63

answered Nov 22, 2009 at 23:20



[Simon_Weaver](#)

146k ● 90 ● 673 ● 713

1 @barett - fixed it. looks like they moved it from 'aspnet' category to 'validation' category which changed the link – [Simon_Weaver](#) Aug 20, 2010 at 19:56

2 Link is broken. Maybe this is the same utility? [codeproject.com/Articles/20271/...](#) – [Josh Noe](#) Feb 15, 2013 at 20:45

That codeproject code is from 2007. Warning, It may be outdated. – [aron](#) Apr 4, 2019 at 1:57



Anatoliy's answer in PHP:

9



```
public static function detectCardType($num)
{
    $re = array(
        "visa"          => "/^4[0-9]{12}(?:[0-9]{3})?$/",
        "mastercard"    => "/^5[1-5][0-9]{14}$/",
        "amex"          => "/^3[47][0-9]{13}$/",
        "discover"      => "/^6(?:011|5[0-9]{2})[0-9]{12}$/",
    );

    if (preg_match($re['visa'], $num))
    {
        return 'visa';
    }
    else if (preg_match($re['mastercard'], $num))
    {
        return 'mastercard';
    }
    else if (preg_match($re['amex'], $num))
    {
        return 'amex';
    }
    else if (preg_match($re['discover'], $num))
    {
        return 'discover';
    }
    else
    {
        return false;
    }
}
```

Share Improve this answer

answered Jun 20, 2015 at 17:58

Follow



angelcool.net

2,546 ● 1 ● 25 ● 26



Compact javascript version

8



```
var getCardType = function (number) {
  var cards = {
    visa: /^4[0-9]{12}(?:[0-9]{3})?$/,
    mastercard: /^5[1-5][0-9]{14}$/ ,
    amex: /^3[47][0-9]{13}$/ ,
    diners: /^3(?:0[0-5]|[68][0-9])[0-9]{11}$/ ,
    discover: /^6(?:011|5[0-9]{2})[0-9]{12}$/ ,
    jcb: /^(?:2131|1800|35\d{3})\d{11}$/
  };
  for (var card in cards) {
    if (cards[card].test(number)) {
      return card;
    }
  }
};
```

Share Improve this answer

answered Mar 25, 2014 at 10:33

Follow



Nick

12k ● 8 ● 45 ● 47



The first numbers of the credit card can be used to approximate the vendor:

8



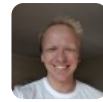
- Visa: 49,44 or 47
- Visa electron: 42, 45, 48, 49
- MasterCard: 51
- Amex:34
- Diners: 30, 36, 38

- JCB: 35

Share Improve this answer

Follow

edited Aug 6, 2014 at 22:09



Gajus

73.5k ● 80 ● 294 ● 469

answered Sep 16, 2008 at 14:56



Shoban

23k ● 8 ● 66 ● 107

These ranges have been updated majorly, Card vendor companies have added way more ranges than mentioned in the post. – [Nagama Inamdar](#) Sep 10, 2019 at 9:44



8



In Card Range Recognition (CRR), a drawback with algorithms that use a series of regex or other hard-coded ranges, is that the BINs/IINs do change over time in my experience. The co-branding of cards is an ongoing complication. Different Card Acquirers / merchants may need you treat the same card differently, depending on e.g. geolocation.

Additionally, in the last few years with e.g. UnionPay cards in wider circulation, existing models do not cope with new ranges that sometimes interleave with broader ranges that they supersede.

Knowing the geography your system needs to cover may help, as some ranges are restricted to use in particular countries. For example, ranges 62 include some AAA sub-ranges in the US, but if your merchant base is outside the US, you may be able to treat all 62 as

UnionPay.

You may be also asked to treat a card differently based on merchant location. E.g. to treat certain UK cards as debit domestically, but as credit internationally.

There are very useful set of rules maintained by one major Acquiring Bank. E.g.

<https://www.barclaycard.co.uk/business/files/BIN-Rules-EIRE.pdf> and

<https://www.barclaycard.co.uk/business/files/BIN-Rules-UK.pdf>. (Valid links as of June 2017, thanks to the user

who provided a link to updated reference.) But be aware of the caveat that, while these CRR rules may represent the Card Issuing universe as it applies to the merchants acquired by that entity, it does not include e.g. ranges identified as CUP/UPI.

These comments apply to magnetic stripe (MagStripe) or PKE (Pan Key Entry) scenarios. The situation is different again in the ICC/EMV world.

Update: Other answers on this page (and also the linked WikiPedia page) have JCB as always 16 long. However, in my company we have a dedicated team of engineers who certify our POS devices and software across multiple acquiring banks and geographies. The most recent Certification Pack of cards this team have from JCB, had a pass case for a 19 long PAN.

Share Improve this answer

edited Jan 24, 2018 at 14:54

Follow

Hi @CaiqueOliveira, see updated links. Thanks to mac9416 who provided a link to updated BIN-Rules reference.

– MikeRoger Jun 16, 2017 at 9:30

1 Thanks @mac9416, for the updated BIN-Rules reference.

– MikeRoger Jun 16, 2017 at 9:30



8



Here is a php class function returns CCtype by CCnumber.

This code not validates the card or not runs Luhn algorithm only try to find credit card type based on table in [this page](#). basicly uses CCnumber length and CCcard prefix to determine CCcard type.

```
<?php
class CreditcardType
{
    public static $creditcardTypes = [
        [
            'Name' => 'American Express',
            'cardLength' => [15],
            'cardPrefix' => ['34', '37'],
        ], [
            'Name' => 'Maestro',
            'cardLength' => [12, 13, 14, 15, 16, 17, 18],
            'cardPrefix' => ['5018', '5020', '5038', '5042', '5043', '5044', '5047', '5048', '5049', '5050', '5051', '5052', '5053', '5054', '5055', '5056', '5057', '5058', '5059', '5060', '5061', '5062', '5063', '5064', '5065', '5066', '5067', '5068', '5069', '5070', '5071', '5072', '5073', '5074', '5075', '5076', '5077', '5078', '5079', '5080', '5081', '5082', '5083', '5084', '5085', '5086', '5087', '5088', '5089', '5090', '5091', '5092', '5093', '5094', '5095', '5096', '5097', '5098', '5099'],
        ], [
            'Name' => 'Mastercard',
            'cardLength' => [16],
            'cardPrefix' => ['51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96', '97', '98', '99'],
        ]
    ]
}
```



```

    ], [
      'Name' => 'Visa',
      'cardLength' => [13, 16],
      'cardPrefix' => ['4'],
    ], [
      'Name' => 'JCB',
      'cardLength' => [16],
      'cardPrefix' => ['3528', '3529', '353', '3
'358'],
    ], [
      'Name' => 'Discover',
      'cardLength' => [16],
      'cardPrefix' => ['6011', '622126', '622127
'62213', '62214', '62215', '62216', '62217', '62218', '
'6224', '6225', '6226', '6227', '6228', '62290', '62291
'622922', '622923', '622924', '622925', '644', '645', '
'65'],
    ], [
      'Name' => 'Solo',
      'cardLength' => [16, 18, 19],
      'cardPrefix' => ['6334', '6767'],
    ], [
      'Name' => 'Unionpay',
      'cardLength' => [16, 17, 18, 19],
      'cardPrefix' => ['622126', '622127', '6221
'62214', '62215', '62216', '62217', '62218', '62219', '
'6225', '6226', '6227', '6228', '62290', '62291', '6229
'622923', '622924', '622925'],
    ], [
      'Name' => 'Diners Club',
      'cardLength' => [14],
      'cardPrefix' => ['300', '301', '302', '303
'], [
      'Name' => 'Diners Club US',
      'cardLength' => [16],
      'cardPrefix' => ['54', '55'],
    ], [
      'Name' => 'Diners Club Carte Blanche',
      'cardLength' => [14],
      'cardPrefix' => ['300', '305'],
    ], [
      'Name' => 'Laser',
      'cardLength' => [16, 17, 18, 19],
      'cardPrefix' => ['6304', '6706', '6771', '

```

```

    ],
];

public static function getType($CCNumber)
{
    $CCNumber = trim($CCNumber);
    $type = 'Unknown';
    foreach (CreditcardType::$creditcardTypes as $card) {
        if (! in_array(strlen($CCNumber), $card['lengths'])) {
            continue;
        }
        $prefixes = '/^(' . implode('|', $card['prefixes']) . ')';
        if (preg_match($prefixes, $CCNumber) == 1) {
            $type = $card['name'];
            break;
        }
    }
    return $type;
}
}

```

Share Improve this answer

edited Jan 8, 2020 at 15:28

Follow



William Desportes

1,698 ● 3 ● 26 ● 37

answered Aug 13, 2013 at 18:29



ismail

377 ● 4 ● 8

Note that this does not handle the new Mastercard prefixes (2221–2720) that were introduced in 2017. – [Costa](#) Dec 8, 2021 at 6:18



Do not try to detect credit card type as part of processing a payment. You are risking of declining valid transactions.

6

If you need to provide information to your payment processor (e.g. PayPal credit card object requires to name the [card type](#)), then guess it from the least information available, e.g.

```
$credit_card['pan'] = preg_replace('/^[^0-9]/', '', $credit_card['pan']);
$inn = (int) mb_substr($credit_card['pan'], 0, 2);

// @see
http://en.wikipedia.org/wiki/List_of_Bank_Identification_Numbers
if ($inn >= 40 && $inn <= 49) {
    $type = 'visa';
} else if ($inn >= 51 && $inn <= 55) {
    $type = 'mastercard';
} else if ($inn >= 60 && $inn <= 65) {
    $type = 'discover';
} else if ($inn >= 34 && $inn <= 37) {
    $type = 'amex';
} else {
    throw new \UnexpectedValueException('Unsupported card type');
}
```

This implementation (using only the first two digits) is enough to identify all of the major (and in PayPal's case all of the supported) card schemes. In fact, you might want to skip the exception altogether and default to the most popular card type. Let the payment gateway/processor tell you if there is a validation error in response to your request.

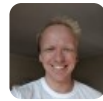
The reality is that your payment gateway [does not care about the value you provide](#).

Share Improve this answer

edited Aug 6, 2014 at 21:57

Follow

answered Feb 26, 2014 at 7:05



Gajus

73.5k ● 80 ● 294 ● 469

1 This is simply untrue. I know of 3 different providers that DO require card types to be passed in, and if you do not pass it in, the transaction will fail. – [Ed DeGagne](#) Apr 15, 2014 at 12:52 ✎

3 @EdDeGagne - "does not care what value" is not the same as "does not care if passed in". – [Quentin Skousen](#) Aug 6, 2014 at 19:56

Where did I specify either? I simply mentioned that there are providers in use that require YOU to pass in the CC type, nothing more. – [Ed DeGagne](#) Aug 18, 2014 at 19:28

you cant simplify this complex issue, but usually payment providers do not require for you to suggest card type, they have their own method to detect – [Janos Szabo](#) Jun 15, 2016 at 12:16



Swift 2.1 Version of Usman Y's answer. Use a print statement to verify so call by some string value

5



```
print(self.validateCardType(self.creditCardField.text))

func validateCardType(testCard: String) -> String
{
    let regVisa = "^4[0-9]{12}([0-9]{3})?$"
    let regMaster = "^5[1-5][0-9]{14}$"
    let regExpress = "^3[47][0-9]{13}$"
    let regDiners = "^3(?:0[0-5]|[68][0-9])[0-9]{11}$"
    let regDiscover = "^6(?:011|5[0-9]{2})[0-9]"
```

```

{12}}$"
    let regJCB = "^(?:2131|1800|35\\d{3})\\d{11}$"

    let regVisaTest = NSPredicate(format: "SELF
MATCHES %@", regVisa)
    let regMasterTest = NSPredicate(format: "SELF
MATCHES %@", regMaster)
    let regExpressTest = NSPredicate(format: "SELF
MATCHES %@", regExpress)
    let regDinersTest = NSPredicate(format: "SELF
MATCHES %@", regDiners)
    let regDiscoverTest = NSPredicate(format:
"SELF MATCHES %@", regDiscover)
    let regJCBTest = NSPredicate(format: "SELF
MATCHES %@", regJCB)

    if regVisaTest.evaluateWithObject(testCard){
        return "Visa"
    }
    else if
regMasterTest.evaluateWithObject(testCard){
        return "MasterCard"
    }

```

Share Improve this answer

edited Apr 7, 2016 at 18:07

Follow

answered Apr 7, 2016 at 17:54



Daisy R.

633 ● 1 ● 7 ● 19



4

Stripe has provided this fantastic **javascript** library for card scheme detection. Let me add few code snippets and show you how to use it.



Firstly Include it to your web page as



```
<script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/jquery.p
" ></script>
```

Secondly use the function cardType for detecting the card scheme.

```
$(document).ready(function() {
    var type = $.payment.cardType("4242
4242 4242 4242"); //test card number
    console.log(type);
});
```

Here are the reference links for more examples and demos.

1. [Stripe blog for jquery.payment.js](#)
2. [Github repository](#)

Share Improve this answer

answered Apr 29, 2015 at 7:36

Follow



Nagama Inamdar

2,857 ● 22 ● 40 ● 50



In swift you can create an enum to detect the credit card type.

4



```
enum CreditCardType: Int { // Enum which
encapsulates different card types and method to
find the type of card.
```



```
case Visa
case Master
case Amex
case Discover

func validationRegex() -> String {
    var regex = ""
    switch self {
    case .Visa:
        regex = "^4[0-9]{6,}$"

    case .Master:
        regex = "^5[1-5][0-9]{5,}$"

    case .Amex:
        regex = "^3[47][0-9]{13}$"

    case .Discover:
        regex = "^6(?:011|5[0-9]{2})[0-9]{12}$"
    }

    return regex
}

func validate(cardNumber: String) -> Bool {
    let predicate = NSPredicate(format: "SELF MATCHES %@", validationRegex())
    return
    predicate.evaluateWithObject(cardNumber)
}

// Method returns the credit card type for given
card number
```

Call the method

CreditCardType.cardTypeForCreditCardNumber("#card number") which returns CreditCardType enum value.

Share Improve this answer

answered Jun 1, 2016 at 5:15

Follow



Vidyalaxmi

309 ● 1 ● 5



A javascript improve of @Anatoliy answer

4



```
function getCardType (number) {
  const numberFormatted = number.replace(/\D/g, '')
  var patterns = {
    VISA: /^4[0-9]{12}(:[0-9]{3})?$/,
    MASTER: /^5[1-5][0-9]{14}$/,
    AMEX: /^3[47][0-9]{13}$/,
    ELO: /^((((636368)|(438935)|(504175)|(451416)|
(636297))\d{0,10})|((5067)|(4576)|
(4011))\d{0,12})$/,
    AURA: /^(5078\d{2})(\d{2})(\d{11})$/,
    JCB: /^(?:2131|1800|35\d{3})\d{11}$/,
    DINERS: /^3(?:0[0-5]|[68][0-9])[0-9]{11}$/,
    DISCOVERY: /^6(?:011|5[0-9]{2})[0-9]{12}$/,
    HIPERCARD: /^(606282\d{10})(\d{3})?|
(3841\d{15})$/,
    ELECTRON:
/^ (4026|417500|4405|4508|4844|4913|4917)\d+$/,
    MAESTRO:
/^ (5018|5020|5038|5612|5893|6304|6759|6761|6762|6763
DANKORT: /^(5019)\d+$/,
    INTERPAYMENT: /^(636)\d+$/,
    UNIONPAY: /^(62|88)\d+$/,
  }
  for (var key in patterns) {
    if (patterns[key].test(numberFormatted)) {
      return key
    }
  }
}

console.log(getCardType("4539 5684 7526 2091"))
```

Share Improve this answer

edited Jul 8 at 6:00

Follow



Kishan Viramgama

1,162 ● 2 ● 14 ● 26

answered Jul 17, 2020 at 13:52



Emanuel

3,247 ● 1 ● 25 ● 28

Seems like 4508 doesn't belong to Visa electron Source:

bincheck.org/visa-visa-electron?page=1

freebinchecker.com/VISA-electron-debit-card-bank

– [igauravsehrawat](#) Oct 11, 2021 at 10:04

Please update MasterCard RegEx to accommodate the contemporary changes. – [Nicholas Mberev](#) Jan 2, 2022 at 9:41



3



My solution with jQuery:

```
function detectCreditCardType() {
    var type = new Array;
    type[1] = '^4[0-9]{12}(?:[0-9]{3})?$';           //
    visa
    type[2] = '^5[1-5][0-9]{14}$';                   //
    mastercard
    type[3] = '^6(?:011|5[0-9]{2})[0-9]{12}$';       //
    discover
    type[4] = '^3[47][0-9]{13}$';                   //
    amex

    var ccnum =
    $('<div>creditcard').val().replace(/[\^\\d.]/g, '');
    var returntype = 0;

    $.each(type, function(idx, re) {
        var regex = new RegExp(re);
        if(regex.test(ccnum) && idx>0) {
            returntype = idx;
        }
    });
}
```

```
return returntype;  
}
```

In case 0 is returned, credit card type is undetected.

"creditcard" class should be added to the credit card input field.

Share Improve this answer

answered Jul 7, 2014 at 16:13

Follow



[ZurabWeb](#)

1,368 ● 1 ● 13 ● 21

1 Variation of existing answers. – [Gajus](#) Aug 6, 2014 at 21:58

1 Yes, I used the code from the above answers, IMPROVED it and posted it here. Thanks for the downvote... – [ZurabWeb](#) Aug 7, 2014 at 15:09

3 You should have (a) suggested this as an improvement to the existing code, (b) written the appropriate contributions, or (c) reference the sources that you have used to write the regular expressions. – [Gajus](#) Aug 8, 2014 at 9:29

1 Gajus, I believe I helped the community the way I could at that moment, please stop telling me I should've done something for someone. I did what I though could've been helpful. – [ZurabWeb](#) Aug 8, 2014 at 15:00



3



I searched around quite a bit for credit card formatting and phone number formatting. Found lots of good tips but nothing really suited my exact desires so I created [this bit of code](#). You use it like this:



```
var sf = smartForm.formatCC(myInputString);  
var cardType = sf.cardType;
```

Share Improve this answer

answered Jul 30, 2015 at 21:35

Follow



ShadeTreeDeveloper

1,573 ● 2 ● 12 ● 21



3



Swift 5+

```
extension String {  
  
    func isMatch(_ Regex: String) -> Bool {  
  
        do {  
            let regex = try  
                NSRegularExpression(pattern: Regex)  
            let results = regex.matches(in: self,  
range: NSRange(self.startIndex..., in: self))  
            return results.map {  
                String(self[Range($0.range, in:  
self)!])  
            }.count > 0  
        } catch {  
            return false  
        }  
  
    }  
  
    func getCreditCardType() -> String? {  
  
        let VISA_Regex = "^4[0-9]{6,}$"  
        let MasterCard_Regex = "^5[1-5][0-9]  
{5,}|222[1-9][0-9]{3,}|22[3-9][0-9]{4,}|2[3-6][0-  
9]{5,}|27[01][0-9]{4,}|2720[0-9]{3,}$"  
        let AmericanExpress_Regex = "^3[47][0-9]  
{5,}$"  
        let DinersClub_Regex = "^3(?:0[0-5]|[68]"
```

```

[0-9])[0-9]{4,}$"
        let Discover_Regex = "^6(?:011|5[0-9]{2})
[0-9]{3,}$"
        let JCB_Regex = "^(?:2131|1800|35[0-9]{3})
[0-9]{3,}$"

        if self.isMatch(VISA_Regex) {
            return "VISA"
        } else if self.isMatch(MasterCard_Regex) {

```

Use.

```
"1234123412341234".getCreditCardType()
```

Share Improve this answer

answered Nov 20, 2020 at 4:37

Follow



[Saharat Sittipanya](#)

321 ● 2 ● 7



2



```

// abobjects.com, parvez ahmad ab bulk mailer
use below script

function isValidCreditCard2(type, ccnum) {
    if (type == "Visa") {
        // Visa: length 16, prefix 4, dashes
        optional.
        var re = /^4\d{3}?\d{4}?\d{4}?\d{4}$/;
    } else if (type == "MasterCard") {
        // Mastercard: length 16, prefix 51-55,
        dashes optional.
        var re = /^5[1-5]\d{2}?\d{4}?\d{4}?
\d{4}$/;
    } else if (type == "Discover") {
        // Discover: length 16, prefix 6011,
        dashes optional.
        var re = /^6011?\d{4}?\d{4}?\d{4}$/;
    } else if (type == "AmEx") {
        // American Express: length 15, prefix

```

34 or 37.

```
var re = /^3[4,7]\d{13}$/;  
} else if (type == "Diners") {  
    // Diners: length 14, prefix 30, 36, or
```

38.

```
var re = /^3[0,6,8]\d{12}$/;  
}  
if (!re.test(ccnum)) return false;  
return true;  
/*  
// Remove all dashes for the checksum  
checks to eliminate negative numbers  
ccnum = ccnum.split("-").join("");  
// Checksum ("Mod 10")  
// Add even digits in even length strings  
or odd digits in odd length strings.  
var checksum = 0;  
for (var i=(2-(ccnum.length % 2));  
i < ccnum.length; i+=2) {
```

Share Improve this answer

answered Jul 17, 2012 at 19:09

Follow



Parvez

69 ● 1

The question is about the *algorithm* to check a credit card, not a specific implementation. What does this code do?

– [Emil Vikström](#) Oct 11, 2012 at 4:35



Just a little spoon feeding:

2



```
$("#CreditCardNumber").focusout(function () {
```

```
var regVisa = /^4[0-9]{12}(?:[0-9]{3})?$/;  
var regMasterCard = /^5[1-5][0-9]{14}$/;  
var regAmex = /^3[47][0-9]{13}$/;  
var regDiscover = /^6(?:011|5[0-9]{2})[0-9]{12}$/;
```

```

        if (regVisa.test($(this).val())) {
            $("#CCImage").html("<img height='40px'
src='@Url.Content("~/images/visa.png")'>");

        }

        else if
(regMasterCard.test($(this).val())) {
            $("#CCImage").html("<img height='40px'
src='@Url.Content("~/images/mastercard.png")'>");

        }

        else if (regAmex.test($(this).val())) {

            $("#CCImage").html("<img height='40px'
src='@Url.Content("~/images/amex.png")'>");

        }
        else if (regDiscover.test($(this).val()))
        {

            $("#CCImage").html("<img height='40px'
src='@Url.Content("~/images/discover.png")'>");

        }
        else {
            $("#CCImage").html("NA");
        }
    }
}

```

Share Improve this answer

answered Jan 31, 2014 at 18:27

Follow



Pinch

4,207 ● 8 ● 42 ● 61



2

Here is an example of some boolean functions written in Python that return `True` if the card is detected as per the function name.



```

def is_american_express(cc_number):
    """Checks if the card is an american express.

```




If us billing address country code, & is_amex, use vpos

https://en.wikipedia.org/wiki/Bank_card_number#cite_GenCardFeatures-3

```
:param cc_number: unicode card number
"""

    return bool(re.match(r'^3[47][0-9]{13}$',
cc_number))

def is_visa(cc_number):
    """Checks if the card is a visa, begins with 4
and 12 or 15 additional digits.
:param cc_number: unicode card number
"""

    # Standard Visa is 13 or 16, debit can be 19
    if bool(re.match(r'^4', cc_number)) and
len(cc_number) in [13, 16, 19]:
        return True

    return False

def is_mastercard(cc_number):
    """Checks if the card is a mastercard. Begins
with 51-55 or 2221-2720 and 16 in length.
:param cc_number: unicode card number
"""

    if len(cc_number) == 16 and
cc_number.isdigit(): # Check digit, before cast
to int
        return bool(re.match(r'^5[1-5]',
cc_number)) or int(cc_number[:4]) in range(2221,
```

Share Improve this answer

answered Apr 27, 2016 at 7:12

Follow



radtek

36.1k ● 13 ● 148 ● 113



1



The first six digits of a card number (including the initial MII digit) are known as the [issuer identification number](#) (IIN). These identify the card issuing institution that issued the card to the card holder. The rest of the number is allocated by the card issuer. The card number's length is its number of digits. Many card issuers print the entire IIN and account number on their card.

Based on the above facts I would like to keep a snippet of **JAVA** code to identify card brand.

Sample card types

```
public static final String AMERICAN_EXPRESS =  
    "American Express";  
public static final String DISCOVER = "Discover";  
public static final String JCB = "JCB";  
public static final String DINERS_CLUB = "Diners  
Club";  
public static final String VISA = "Visa";  
public static final String MASTERCARD =  
    "MasterCard";  
public static final String UNKNOWN = "Unknown";
```

Card Prefixes

```
// Based on  
http://en.wikipedia.org/wiki/Bank_card_number#Issuer  
public static final String[]  
    PREFIXES_AMERICAN_EXPRESS = {"34", "37"};
```

```

public static final String[] PREFIXES_DISCOVER =
{"60", "62", "64", "65"};
public static final String[] PREFIXES_JCB =
{"35"};
public static final String[] PREFIXES_DINERS_CLUB
= {"300", "301", "302", "303", "304", "305",
"309", "36", "38", "39"};
public static final String[] PREFIXES_VISA =
{"4"};
public static final String[] PREFIXES_MASTERCARD =
{
    "2221", "2222", "2223", "2224", "2225",
"2226", "2227", "2228", "2229",
    "223", "224", "225", "226", "227", "228",
"229",
    "23", "24", "25", "26",
    "270", "271", "2720",
    "50", "51", "52", "53", "54", "55"
};

```

Check to see if the input number has any of the given prefixes.

```

public String getBrand(String number) {

String evaluatedType;
if (StripeTextUtils.hasAnyPrefix(number,
PREFIXES_AMERICAN_EXPRESS)) {
    evaluatedType = AMERICAN_EXPRESS;
} else if (StripeTextUtils.hasAnyPrefix(number,
PREFIXES_DISCOVER)) {
    evaluatedType = DISCOVER;
} else if (StripeTextUtils.hasAnyPrefix(number,
PREFIXES_JCB)) {
    evaluatedType = JCB;
} else if (StripeTextUtils.hasAnyPrefix(number,
PREFIXES_DINERS_CLUB)) {
    evaluatedType = DINERS_CLUB;
} else if (StripeTextUtils.hasAnyPrefix(number,
PREFIXES_VISA)) {

```

```

        evaluatedType = VISA;
    } else if (StripeTextUtils.hasAnyPrefix(number,
PREFIXES_MASTERCARD)) {
        evaluatedType = MASTERCARD;
    } else {
        evaluatedType = UNKNOWN;
    }
    return evaluatedType;
}

```

Finally, The Utility method

```

/**
 * Check to see if the input number has any of
the given prefixes.
 *
 * @param number the number to test
 * @param prefixes the prefixes to test against
 * @return {@code true} if number begins with any
of the input prefixes
 */

public static boolean hasAnyPrefix(String number,
String... prefixes) {
    if (number == null) {
        return false;
    }
    for (String prefix : prefixes) {
        if (number.startsWith(prefix)) {
            return true;
        }
    }
    return false;
}

```

Reference

- [Stripe Card Builder](#)

Share Improve this answer

edited Mar 8, 2017 at 9:56

Follow

answered Mar 8, 2017 at 9:49



Anoop M Maddasseri

10.5k ● 3 ● 54 ● 74



1



follow Luhn's algorithm

```
private boolean validateCreditCardNumber(String
str) {

    int[] ints = new int[str.length()];
    for (int i = 0; i < str.length(); i++) {
        ints[i] =
Integer.parseInt(str.substring(i, i + 1));
    }
    for (int i = ints.length - 2; i >= 0; i =
i - 2) {
        int j = ints[i];
        j = j * 2;
        if (j > 9) {
            j = j % 10 + 1;
        }
        ints[i] = j;
    }
    int sum = 0;
    for (int i = 0; i < ints.length; i++) {
        sum += ints[i];
    }
    if (sum % 10 == 0) {
        return true;
    } else {
        return false;
    }

}
```

then call this method

```
EditText mCreditCardNumberEt;  
  
mCreditCardNumberEt.addTextChangedListener(new
```

Share Improve this answer

edited Mar 28, 2018 at 9:51

Follow

answered Mar 28, 2018 at 9:38



gaurav gupta

563 ● 1 ● 6 ● 14



Try this for kotlin. Add Regex and add to the when statement.

1



```
private fun getCardType(number: String): String {  
    val visa = Regex("^4[0-9]{12}(?:[0-9]{3})?"  
$")  
    val mastercard = Regex("^5[1-5][0-9]{14}$")  
    val amx = Regex("^3[47][0-9]{13}$")  
  
    return when {  
        visa.matches(number) -> "Visa"  
        mastercard.matches(number) ->  
"Mastercard"  
        amx.matches(number) -> "American  
Express"  
        else -> "Unknown"  
    }  
}
```

Share Improve this answer

answered Oct 4, 2019 at 18:43

Follow

OhhhThatVarun

4,310 ● 2 ● 33 ● 54



0



The regular expression rules that match the [respective card vendors](#):

- `(4\d{12}(?:\d{3})?)` for VISA.
- `(5[1-5]\d{14})` for MasterCard.
- `(3[47]\d{13})` for AMEX.
- `((?:5020|5038|6304|6579|6761)\d{12}(?:\d\d)?)` for Maestro.
- `(3(?:0[0-5]||[68][0-9])[0-9]{11})` for Diners Club.
- `(6(?:011|5[0-9]{2})[0-9]{12})` for Discover.
- `(35[2-8][89]\d\d\d{10})` for JCB.

Share Improve this answer

edited Aug 6, 2014 at 22:07

Follow



Gajus

73.5k ● 80 ● 294 ● 469

answered Feb 24, 2014 at 20:40



rajan

9 ● 1

I think that regex for JCB is incorrect. All first four digits between 3528 and 3589 should be accepted, but 3570 is not, for example. – [Gabe](#) Sep 16, 2016 at 16:01



0



Another api solution at rapidapi [Bank Card Bin Num Check](#) there are 250K+ issued card type.

Only one GET rest api request and get card issuer info like:

```
{ "bin_number": 535177, "bank": "Finansbank A.S.",  
  "scheme": "MASTERCARD", "type": "Debit",  
  "country": "Turkey" }
```

Share Improve this answer

answered Jan 12, 2023 at 9:32

Follow



[Murat Kucukosman](#)

632 ● 5 ● 11



0



Based on that set of rules :

https://raw.githubusercontent.com/Fivell/credit_card_validations/master/lib/data/brands.yaml provided in this post by Fivell I created that php method :

```
public static function identifyCardType($cardNum  
{  
    $cardTypes = [  
        'visa' => [  
            'rules' => [['length' => [13, 16, 19  
            'options' => ['brand_name' => 'Visa'  
        ],  
        'mastercard' => [  
            'rules' => [['length' => [16], 'pref  
                '2221', '2222', '2223', '2224',  
            '2228', '2229',  
                '223', '224', '225', '226', '227  
            '24', '25', '26', '271', '2720', '51', '52', '53', '  
        ]],
```

```

        'options' => ['brand_name' => 'Maste
    ],
    'amex' => [
        'rules' => [['length' => [15], 'pref
        'options' => ['brand_name' => 'Ameri
    ],
    'diners' => [
        'rules' => [['length' => [14], 'pref
'302', '303', '304', '305', '36', '38']]],
        'options' => ['brand_name' => 'Diner
    ],
    'jcb' => [
        'rules' => [
            ['length' => [15, 16], 'prefixes
'353', '354', '355', '356', '357', '358']],
            ['length' => [15], 'prefixes' =>
            ['length' => [19], 'prefixes' =>
        ],
        'options' => ['brand_name' => 'JCB']
    ],
    'solo' => [
        'rules' => [['length' => [16, 18, 19
'6767']]],
        'options' => ['brand_name' => 'Solo'
    ],
    'maestro' => [
        'rules' => [
            ['length' => [12, 13, 14, 15, 16
=> [
                    '500', '5010', '5011', '5012
'5016', '5017', '5018', '502', '503', '504',
                    '505', '506', '507', '508',
'59', '6010', '6012', '6013', '6014', '6015', '6016'
'602', '603', '604', '605', '6060', '616788', '62183
'62198', '62199', '6220', '622110', '627089', '62709
'6273', '6274', '6275', '6276', '6277', '6278', '627
'6301', '630490', '633857', '63609', '6361', '636392
'637102', '637118', '637187', '637529', '639', '640'
'670', '671', '672', '673', '674', '675', '677', '67
'6763', '6764', '6765', '6766', '6768', '6769', '677
                ]]
            ],
        'options' => ['brand_name' => 'Maest
    ],

```

```

        'electron' => [
            'rules' => [
                ['length' => [16], 'prefixes' =>
'4508', '4844', '4913', '4917']]
            ],
            'options' => ['brand_name' => 'Visa
],
        'unionpay' => [
            'rules' => [
                ['length' => [16, 17, 18, 19], '
'620', '6210', '6212', '6213
'6217', '621977', '622126', '622127', '622128', '622
'62213', '62214', '62215', '
'62220', '62221', '62222', '62223', '62224', '62225'
'62226', '62227', '62228', '
'6225', '6226', '6227', '6228', '6229', '623', '624'
'625', '626', '62702', '6270
'6282', '6283', '6284', '6291', '6292', '632062', '6
'69075'
                ]]
            ],
            'options' => ['brand_name' => 'China
true],
        ],
        'discover' => [
            'rules' => [
                ['length' => [16, 19], 'prefixes
'6011', '622126', '622127',
'62213', '62214', '62215', '62216', '62217', '62218'
'6222', '6223', '6224', '622
'62290', '62291', '622920', '622921', '622922',
'622923', '622924', '622925'
'647', '648', '649', '65'
                ]]
            ],
            'options' => ['brand_name' => 'Disco
],
        'rupay' => [
            'rules' => [
                ['length' => [16], 'prefixes' =>
'6061', '6062', '6063', '606
'6068', '6069', '607', '608'
                ]]
            ],

```

```

        'options' => ['brand_name' => 'RuPay
    ],
    'hipercard' => [
        'rules' => [
            ['length' => [19], 'prefixes' =>
                '384100', '384140', '384160'
'637568', '637599', '637609', '637612'
            ]]
        ],
        'options' => ['brand_name' => 'Hiper
    ],
    'elo' => [
        'rules' => [
            ['length' => [16], 'prefixes' =>
                '401178', '401179', '431274'
'457393', '457631', '457632', '504175', '506699', '5
                '506717', '506718', '50672',
'506724', '506725', '506726', '506727', '506728', '5
                '50674', '50675', '50676', '
'506772', '506773', '506774', '506775', '506776', '5
                '50900', '509013', '50902',
'509035', '509036', '509037', '509038', '509039', '5
                '509051', '509052', '509053'
'509067', '509068', '509069', '509072', '509074', '5
                '509078', '509079', '509080'
'509083', '509085', '636297', '636368', '650031', '6
                '650035', '650036', '650037'
'65004', '650050', '650051', '650405', '650406', '65
                '650409', '65041', '65042',
'650486', '650487', '650488', '650489', '65049', '65
                '650530', '650531', '650532'
'650535', '650536', '650537', '650538', '650541', '6
                '650901', '650921', '650922'
'650925', '650926', '650927', '650928', '650929', '6
                '65095', '65096', '650970',
'650973', '650974', '650975', '650976', '650977', '6
                '65500', '65501', '655021',
            ]]
        ],
        'options' => ['brand_name' => 'Elo']
    ],
    'mir' => [
        'rules' => [
            ['length' => [16], 'prefixes' =>

```

```

                '2200', '2201', '2202', '220
            ]]
        ],
        'options' => ['brand_name' => 'Mir']
    ],
    'dankort' => [
        'rules' => [
            ['length' => [16], 'prefixes' =>
        ],
        'options' => ['brand_name' => 'Danko
    ],
    'switch' => [
        'rules' => [
            ['length' => [16, 18, 19], 'pref
                '4903', '4905', '4911', '493
'6333', '6759'
            ]]
        ],
        'options' => ['brand_name' => 'Switc
    ]

];

foreach ($cardTypes as $type => $data) {
    foreach ($data['rules'] as $rule) {
        if (in_array(strlen($cardNumber), $r
            foreach ($rule['prefixes'] as $p
                if (strpos($cardNumber, $pre
                    return isset($data['opti
$data['options']['brand_name'] : null;
        }
    }
}
}
}

return 'Unknown';
}

```

 Run code snippet

[Expand snippet](#)

Share Improve this answer

answered Jan 16 at 11:22

Follow



Hugo Trial

396 ● 5 ● 15



Highly active question. Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.