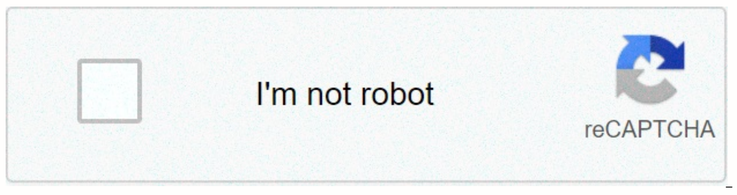

Continue

Vba excel format date variable

Here is a list of most of the characters that can be used to set date and time formats: CharactersExampleDescription m2Month (numeric without zeros) mm02Month (numeric with zeros) mmmFebMonth (short text) mmmmFebruaryMonth (full-length text) d7Day (full-length text) d7Day (full-length text) d7Day (full-length text) dd07Day (numeric text) with zeros dddTueDay (short text) ddddTuesdayDays (full-length text) yy12Year (last 2 digits) yyyy2012Year (4 digits) h8Hour zero-free (0-23) hh08Hours with zeros (00-00-00-00-0 23) n3Minutes without zeros (0-59) nn03Minutes with zeros (00-59) s8Seconds without zeros (0-59) ss08Seconds with zeros (00-59) AM/PMDisplay AM/PM And here are some examples of date and time formats: Sub date_and_time() 'Now > returns the current date and time (02.07.2012 09:09:02) date_test ? Now() 'Returns: 02.07.12 Range(A1) ? Format(date_test, mm.dd.yy) 'Returns: Range of February 7, 2012(A2) - Format(date_test , d mmmm yyyy) 'Returns: February 7, 2012 Range(A3) ? Format(date_test, mmmm j, yyyy) 'Returns: Mar 07 Range(A4) ? Format(date_test, ddd dd) 'Returns: February-12(Range A6) - Format(date_test, mmmm-ya) 'Returns: 02.07.2012 09:09 Range(A7) to Format(date_test, mm.dd.yyyy hh:mm) 'Returns: 2.7.12 9:09 AM Range(A8) to Format(date_test, m.d.yy h:mm AM/PM) 'Returns: 9H09 Range(A9) - Format(date_test, h-Hmm) Final Beginner VBA User here. I think this might be a very easy question to answer, but I can't find the solution here anywhere else, so I apologize if that's the case. I'm writing a very short piece of code that inserts the last business day into a column where it is not already present and moves to the next column and runs the same process where it is present. The code's downstairs. Note: I simplified this but removed the else condition. I want to set the date format for the last business day variable to be dd/mm/yyyy - at the time its including hh/mm/ss which is causing the if function to not function properly. How can I declare the conditions of this variable? Sub Dateextend4() Dim lastbusinessday As Date If Weekday(Now()) to 2 Then lastbusinessday ? (Now() - 3) Else lastbusinessday á (Now() - 1) End If Sheets(pb CDS). Select Range (b13). Select Selection.End(xlDown). Select If ActiveCell.Offset(0, 0). Value < lastbusinessday Then ActiveCell.Offset(1, 0) a format(lastbusinessday, mm/dd/yyyy) Else End If End Sub Thanks Excel gives you several options for formatting dates. In addition to the various built-in date formats that exist, you can create custom date formats. Although the process of manually applying a date format is not very complicated, there are some circumstances in which you may want to create macros that format dates. This may be the case if, for example: You use a particular date format constantly and want to be able to such a format without having to do everything manually; or frequently formats cells or cell ranges in a particular way, and the formatting rules that are applied include date formats. Date. if you're interested in understanding how you can use Visual Basic for Applications for date formatting purposes, you've found the right place. When working in Visual Basic for Applications, there are a few different properties and functions that you can use to format a date. The following 3 are commonly used: VBA Format function. The Range.NumberFormatLocal property. The Range.NumberFormat property. This particular Excel tutorial focuses on the last item in the previous list (the Range.NumberFormat property). I can cover the Format function and the Range.NumberFormatLocal property in future blog posts. If you would like to be informed each time you post new content to power Spreadsheets, be sure to register for our Newsletter by entering your email address below. In addition to explaining the Range.NumberFormat property, I explain the different date format codes that you can use and introduce 25 date formatting examples using VBA. You can use the following detailed table of contents to navigate to the section of this tutorial that interests you most. Before presenting the NumberFormat property in more detail, let's start by taking a look at the sample file that accompanies this Excel tutorial: Formatting Dates with Excel VBA: Example For this Excel tutorial, I use an Excel workbook that contains the full match schedule of the Brazil 2014 World Cup. This Excel vba date formatting tutorial is accompanied by an Excel workbook that contains the data and some versions of the macros I explain below. You can get immediate free access to this sample workbook by subscribing to the Power spreadsheet newsletter. Notice how the first column of the table contains dates: these are the dates I format throughout this tutorial. However, since the focus of this macro is on formatting dates using VBA, we need a Sub procedure. The following image shows the basic structure of the macro (Format_Dates) that I use to format these dates. The macro has only one statement: Selection.NumberFormat a m/d/yy; So, before we start showing examples of how you can format dates using VBA, let's look at this statement. For these purposes, you simply need to understand... The Range.NumberFormat property and how to format an Excel date using VBA as mentioned at the beginning of this Excel tutorial, you can generally use the Range.NumberFormat property to format the dates. The Range.NumberFormat property sets a Variant value. This value represents the numeric format code of the relevant Range object. For these purposes, the Range object is usually a single cell or a range of cells. Strictly speaking, in addition to setting the NumberFormat property value, you can also return the settings property. As Excel authority John Walkenbach explains in Excel VBA Programming for Dummies, the NumberFormat property is a read/write property. However, if you are reading this Excel tutorial, you probably want to modify the property, property, Read. Therefore, this guide focuses on how to change the NumberFormat property, not how to read it. However, you can easily examine the numeric formatting of a cell or range of cells. I explain how you can read a property value in this tutorial. In such cases, if (i) you select a range of cells and (ii) all cells do not share the same format, the Range.NumberFormat property returns Null. NumberFormat is just one of many properties (almost 100 per count) of the Range object. As explained in Excel Macros for dummies, once you have selected a range of cells (as the sample Format_Dates macro does), you can use any of the Range properties to manipulate the cells. This Excel tutorial is quite specific. The only property of the Range object that I cover in this blog post is NumberFormat. In fact, I only explain (in great detail) one of the applications of the NumberFormat property: formatting dates with VBA. I can cover other properties of the Range object, or other applications of the NumberFormat property, in future tutorials. If you would like to receive an email each time I post new material to Power spreadsheets, be sure to subscribe to our newsletter by entering your email address below: Range.NumberFormat Property Syntax The Range.NumberFormat property syntax is relatively simple: expression. NumberFormat In this case, expression means the Range object or a variable that represents this object. The sample Format_Dates macro shown above uses the Application.Selection property, which returns the selected object. Typically, you can use the example Format_Dates macro frame as long as the selection is a range of cells. Therefore, the sample macro uses the following version of the preceding syntax: Selection.NumberFormat You can use another expression instead of Selection. What matters, as I mentioned earlier, is that the expression represents a Range object. Whenever you want to modify the value of a property, you must do the following: #1: Determine whether the property you are working with uses arguments and, if that is the case, determine which argument you want to use. These arguments specify the value that the property takes. #2: Use an equal sign to separate the property name from the property value. As shown in the examples in this tutorial, if you are implementing the NumberFormat property using the framework structure of the Format_Dates sample macro, the argument values are usually surrounded by double quotation marks ( ). Therefore, if you are setting the value of the NumberFormat property, you can use the following syntax: expression. NumberFormat argument_value In other words, to change the current setting of the NumberFormat property, a statement is used that includes the 3 elements Item #1: A reference to the NumberFormat property. #2 of elements: the equals sign (or). Item #3: The new value of the NumberFormat property, surrounded by double quotation marks ( ). As I explained above, the property determines the numeric format code of a Range object. Therefore, in order to format a date using VBA, you need to understand... Date format codes: The Range.NumberFormat property arguments as explained in Microsoft Den center: The format code is the same string as the Format Codes option in the Format Cells dialog box. This is a bite, so let's break down the statement and process into different parts to understand how you can know which formatting code you want to apply. More precisely, you can find the string that represents a particular format code in the Format Cells dialog box in the following 5 simple steps. This process is especially useful if you do not know the formatting code you want to apply. Generally, as you become familiar with numeric format codes, you can create macros that format dates without having to go through this every time. For these purposes, see the introduction to the date format codes below. Step #1: Go to the Number tab of the Format Cells dialog box you can access the Format Cells dialog box using any of the following methods: Method #1: Click the launcher in the dialog box in the lower-right corner of the Number command group on the Home ribbon tab. Method #2: Go to the Home tab of the ribbon, expand the Number Format drop-down list, and select More Number Formats. Method #3: Use the keyboard shortcut Ctrl + 1. Regardless of which of the above methods you use, Excel displays the Format Cells dialog box. If you use the #1 or #2 method, Excel displays the Number tab, as in the previous image. This is the one you need to find the date format codes. However, if you use the #3 method (keyboard shortcut), Excel can show you a tab other than the Number tab (as shown above). In this case, simply go to the Number tab. Step #2: Select the Date Category Since you are interested in date formatting codes, choose Date from the Category list box on the left side of the Format Cells dialog box. Step #3: Choose the type of date format whose format code you want Once you have selected the Date category, Excel displays the built-in date format types within the Type box on the right side of the Format Cells dialog box. This allows you to select from several different date format types. For example, in the image above, I select option 14-Mar-12: Step #4: Select the custom category Once you have selected the type of date format you are interested in, click Custom under the Category list box on the right side of the Format Cells dialog box. Step #5: Get date format code One which has completed the previous 4 steps, Excel displays the date format code that corresponds to the type of date format that you selected in step #3 step. This formatting code is displayed in the Type box in the upper-right section of the Format Cells dialog box. The date format code shown in the previous example is [$-en-US]d-mmm-yy;-. This format code corresponds to option 14-Mar-12 with the English (United States) locale that step #3. Once you have this date format code, you can return to your VBA code and use it as an argument to the Range.NumberFormat property. To see how this works in practice, let's go back to the World Cup calendar above. If you want to apply the formatting shown above, VBA code looks like this: as below screenshot shown, you can achieve the same date formatting effect without the first part of the date format code that references the locale. That means you can delete [$-en-US]. However, for the time being, I leave it inside. For this example, I modify the format of the dates that appear in the sample table. Supposing, before applying this new version of the Format_Dates macro, all dates are formatted as long date, as following screenshot shown: Before running the Format_Dates macro, I select the cell I want to format. In this case, I choose the date in the first row of the table. This corresponds to 12 June 2014, which is the date of the match between Brazil and Croatia. Once I run the previous version of the Format_Dates macro, the date format changes to the following: The 5-step method for finding the date format codes described above can be useful in some situations. However, Excel date format codes follow some general rules. If you know them, you don't have to go through the whole process described above every time you want to create a macro that forms the dates. Let's take a look at these general rules and some additional examples: Date Formatting Codes in Excel and VBA: General Guidelines The date formatting codes you can use to format a date with VBA appear in the following table. As shown in the following sections, you can use these codes to create different types of date formats to use in VBA code. The formatting code applies to ToFormat CodeDescription As seen in practice MonthmMonth is displayed as a number. Does not include a 0.1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 MonthmmDisplays month as a number. If the month is between January (month 1) and September (month 9), it includes an initial name of 0.01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12 MonthmmmMonth is abbreviated. Jan, Feb, Sea, April, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec MonthmmRemose month name is displayedJanio, F, March, April, May, June, July, August, September, October, November, December MonthmmmMMióso is shown the first letter of the name of the mesJ, F, M, A, M, J, D Day (Number)dThe day number is displayed without an interlock 0.1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 Day (Number)ddThe day number is displayed. For days 1 to 9, an initial 0.01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 Day (day of the week)dddThe day of the week is abbreviatedMon, Sea, Wed, Thu, Fri, Sat, Sun Day ()ddddThe full name is day of the week is abbreviatedMon, Sea, Wed, Thu, Fri, Sat, Day of the Sun ()ddddThe full name is day of the week is abbreviatedMon, Sea, Wed, Thu, Fri, Sat, Day of the Sun ()ddddThe full name is day of the week is abbreviatedMon, Sea, Wed, Thu, Fri, Sat, Day of the Sun ()ddddThe full name is day of the week is abbreviatedMon, Tue, Thu, Fri, Sat, Day of the Sun ()ddddThe full name is day of the week is abbreviatedMon, Tue, Thu, Fri, Sat, Day of the Sun ()ddddThe full name is day of the week is abbreviatedMon, Tue, Thu, Fri, Sabbath ()ddddThe full name is day of the week is abbreviatedMon, Tue, Thu, Fri, Sat, Day of the Sun ()ddddThe full name is day of the week is abbreviatedMon, Tue, Thu, Fri, Sat, , Sea, Wed, Thu, Fri, Sat, Sun Day ()ddddThe full name is the day of the week is abbreviatedMon, Tue, Wed, Thu, Fri, Sat, Sun Day ()ddddThe full name is day of the week is abbreviatedMon, Tue, Wed, Thu, Fri, Sab, Sun Day ()ddddThe full name is the day of the week is abbreviatedMon, , Sat, Sun Day ()ddddThe full name is the day of the week is abbreviatedMon, Tue, Wed, Thu, Fri, Sat, Sat, Sun Day ()ddddThe name name Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday YearsThe last 2 digits of the year are displayed00 to 99 Yearsall of the four digits of the year are displayed1900 to 9999 Note, however, that the Format function (which I mention at the beginning of this Excel tutorial) supports slightly different date format codes than those shown in this table. The following sections show examples of how you can apply all these different options to format dates using VBA. For each situation, I show the following: the VBA code in the Format_Dates macro. The resulting format on one of the dates within the sample table containing the 2014 Brazil World Cup calendar. Let's start taking a look at each of these: Date format with VBA: Show a single date item You can have Excel display only one of the items of a date, regardless of whether it's the month, day, or year. To do this, the Range.NumberFormat property argument should only include the format code of the relevant element. Let's take a look at the different date formats you can get including a single item, and how the corresponding VBA code looks: Format a date to show only the month Using VBA You can display a month using any of the following 5 options: Option #1: Show month as a number (no leading 0) You can format a date in a way that: Only the month is displayed; and the month is displayed as a number without an initial 0. In this case, the format code that is used as an argument to the NumberFormat property is m. The image shows the sample version of Format_Dates macro that does this: Let's go back to the sample table with the 2014 World Cup calendar. I select the second date, which is June 13, 2014 and corresponds to the match between Mexico and Cameroon. The following image shows the results after the Format_Dates macro formats the date. Notice how only the month number (6, corresponding to June) appears. Note, too, that the value of the formula bar remains the same. The only thing that changes is the format of the date that is displayed in the cell itself. Option #2: Show the month as a number (with leading 0) This option is similar to the previous one. More particular: Only the month is displayed; and the month is displayed as a number. However, in this particular case, the month is displayed with an initial 0. In other words, if the corresponding month is between January (month 01) and September (month 09), an initial 0 is added. For these purposes, the VBA code behind the Format_Dates macro looks like this: This particular macro applies to the third date in the 2014 World Cup calendar. The date is June 13, 2014. The teams that play are Spain and Holland. Once the macro is applied to Format_Dates, the date is seen as follows in the cell (where it changes formula bar (where the value remains the same): Option #3: Show month as a 3-letter abbreviation If you decide to implement option, the month name is displayed as a 3-letter abbreviation. To achieve this, the VBA code of the Format_Dates macro is as follows: Let's continue with the same process of applying the new date formats to the match dates of the Brazil 2014 World Cup. In this case, the relevant date is June 13, 2014. The match played is between Chile and Australia. The following image shows the appearance of the cell after applying the new formatting. As in the previous cases, the value of the date itself (as shown in the formula bar) does not change. Option #4: Display the full name of the month If you want to display the full name of the month that corresponds to a date (not just its abbreviation), the following version of the Format_Dates macro helps: To apply this format to a date in the Brazil 2014 World Cup calendar, I select the corresponding cell. In this case, the date is June 14, 2014 and corresponds to the party between Colombia and Greece. The results of applying the new version of the Format_Dates macro are shown in the following screenshot: Option #5: Show the first letter of the month The fifth way you can show only the month when formatting a date with VBA is to display (only) the first letter of the relevant month. In this case, the macro Format_Dates looks like this: This macro applies to June 14, 2014. This date corresponds to the match between Uruguay and Costa Rica. The results of running the new macro are shown in the following image: Format a date to show only the day number using VBA Just as you can use VBA to format a date in a way that only shows the month, you can do the same during the day. In other words, you can use Visual Basic for Applications to format a date and have Excel display only the day. The following 2 sections show how you can modify Format_Dates macro example so that only the day (number) is displayed when the date is formatted. Option #1: Show

day number without leading 0 If you want Excel to display the day number without an initial 0 while using VBA, you can use the following version of the sample Format_Dates macro: the date to which this macro applies in the sample workbook is June 14, 2014. In this case, the relevant match is the one between England and Italy. The results of applying the Format_Dates macro are shown in the following image: Option #2: Show day number with leading 0 You can format dates so that Excel adds an initial 0 as long as the day is only 1 digit long (1 to 9). The next version of the Format_Dates macro accomplishes this: If I continue to go down the match schedule of the 2014 Brazil World Cup (as before), this version of the Format_Dates macro would apply to June 14, 2014. This date corresponds to the match Ivory Coast and Japan. However, since this day (14) does not require a leading 0, the result of applying the new version of the Format_Dates macro would be the same as obtained previously for the date of the match between England and Italy. To see how this date format works as long as the corresponding day is only one digit long, I go further down the sample table to one of the matches played in early July 2014. More precisely, I apply the current version of Format_Dates macro as of July 1, 2014. The match to which this date corresponds is the one played between Argentina and Switzerland. The following image shows the results of applying the Format_Dates macro to this date. Notice how Excel now adds an initial 0 to the day number in the cell. Formatting a date to display only the day of the week with VBA The previous section shows how you can use VBA to format a date in such a way that only the day number is displayed. You can also format a date so that only the day of the week is displayed. The following 2 sections show 2 ways you can implement this date format using VBA. Option #1: Show the day of the week as a 3-letter abbreviation The first way you can format a date to display the day of the week allows you to have that day of the week displayed as a 3-letter abbreviation. The next version of the macro Format_Dates achieves this: Let's get back to the fight between Ivory Coast and Japan that I reference earlier and apply this new date format. The date of this match is June 14, 2014. After running the Format_Dates macro, the date looks like this: Option #2: Show full weekday The second way you can format a date to display the day of the week using VBA causes Excel to display the full name of the day of the week. The following version of the Format_Dates macro formates a date in such a way: Let's run this macro in order to format the date of the fight between Switzerland and Ecuador in the match schedule of the Brazil 2014 World Cup. This date, as shown in the image below, is June 15, 2014. Running the Format_Dates macro while this particular cell is active causes the following change in date format: Format a date to display only the year using VBA so far, you have seen how you can format a date using VBA in order to display only the (i) month, (ii) day number or (iii) day of the week. In this section, I show you how to format a date using VBA to show only the year. Let's take a look at the 2 options you have for these purposes: Option #1: Show last 2 digits of the year The first way you can format a date to show only the year results in Excel showing only the last 2 digits of the relevant year. To achieve this date format, you can use the following version of the Format_Dates macro: This date format will be applied to the date 15 of 2014. This corresponds to the World Cup match between France and Honduras. The following image shows the results of running the Format_Dates macro sample while this cell is active: Option #2: Show full year The second way you can format a date to display only the year results in Excel that show the Year. If you want to format a date in such a way using VBA, the following version of macro Format_Dates achieves this result: I apply this date format to the date of the argentina-Bosnia-Herzegovina match. This is June 15, 2014. Once the Format_Dates macro runs, the results are as following screenshot shown: Date format using VBA: Show multiple date items The examples in the previous section explain different ways you can format a date with VBA to display a single item (month, day or year) of that particular date. Having the ability to format a date in such a way that only a single item is displayed is useful in certain scenarios. In addition, once you know the formatting codes that apply to each of the individual elements of a date, you can easily start combining them to create more complex and advanced date formats. In any case, in many cases, you will need to format the dates in such a way that more than 1 item is displayed. In the following sections, I go through some date formats that result in Excel displaying more than 1 item of the relevant date. Although I don't cover all the date formats you can implement, these examples give you an idea of the possibilities you have at your disposal and how you can implement them in your VBA code. All of the following sections follow the same shape and show 2 things: The version of the Format_Dates macro that is applied. The result of running that macro in order to format 1 of the dates of a match in the sample workbook that accompanies this blog post. Date format using VBA: Show m/d/yyyy The next version of the macro Format_Dates formats a date in the format m/d/yyyy. The following image shows the result of applying this format to june 16, 2014. This date corresponds to the match between Germany and Portugal. Date format using VBA: Show m/d To display a date in m/d format, You can use the following macro: When you run this macro and select the cell with the date of the match between Iran and Nigeria (June 16, 2014), the formatted date looks like this: Date format using VBA: Show m/d/y The following macro formats a date to display in m/d/yy format: The result of applying this format, Using the previous Format_Dates macro version, until the date of June 16, 2014 (for the match between Ghana and the US) is shown below: Date format using VBA : Show mm/dd/yy You can format a date to display in mm/dd/yy format using the following version of the Format_Dates macro: When this date format applies to the date of the World Cup match between Belgium and Algeria (17 June 2014), the result is shown in the following image: Date format using VBA: Show d-mmm The next version of the Format_Dates macro causes Excel to display the date in the form d-mmm: The results of running this macro while the World Cup date matches between Brazil and Brazil Brazil selected (June 17, 2014) shown in the following image: Date format with VBA: Show d-mmm-aa The next version of the Format_Dates macro causes Excel to display dates using the form d-mmm-aa: If I choose the cell that shows the date of the match between Russia and Korea (June 17, 2014) before running this version of the Format_Dates , the resulting date format is as follows: Date format using VBA: Show dd-mmm-aa To apply dd-mmm-aa format to a given date, you can use the following version of the sample Format_Dates macro: The following screenshot shows the results of applying this version of the Format_Dates macro to the date Australia played against the Netherlands at the 2014 Brazilian World Cup (June 18, 2014) : Note that, in this particular case, the resulting date format is exactly the same as that of the date of the Russia-Korea match that is immediately above (and is used as an example in the previous section). To understand why this is the case, let's take a look at the date format codes used in each case: Russia vs Korea (June 17, 2014) uses the date format code d-mmm-aa. Australia vs. Netherlands (June 18, 2014) has the date format code dd-mmm-aa. Notice that the only difference between the two format codes is in the way the day is rendered. In the first case, the format code uses d, which displays the day number without an initial 0. In the second case, the format code is dd, which adds an initial 0 as long as the day number has a single digit (between 1 and 9). In this particular situation, the day numbers of both dates (17 and 18) have 2 digits. Therefore, the dd format code does not add an initial 0 to the day number. The result is the one shown above: Both format codes (d-mmm-aa and dd-mmm-aa) result in the same date format when the number of digits in the day is 2 (between 10 and 31). Let's go further down the 2014 Brazil World Cup match schedule to see how the dd-mmm-aa format code adds an initial 0 when applied to a date when the day number has a single digit: The image below shows this. In this particular case, the macro Format_Dates applied to the date 1 July 2014, when Belgium played against the United States. Note, in particular, the initial 0 in the day number (01 instead of 1). Date format using VBA:mmm-yy The next version of the Format_Dates macro allows you to format a date in mmm-ya format: the resulting date format when running this version of the Format_Date macro is as shown in the following image. The formatted date is June 18, 2014, corresponding to the match between Spain and Chile. Date Format Using VBA: mmmm-yy Continuing with date formats that only show month and year, the following version of the Format_Dates macro applies the format code mmmm-aa: The results of running the macro on June 18, 2014 are shown in the following image. In this case, the date corresponds to the World Cup match between Cameroon and Croatia. Format date VBA: mmmm d, yyyy The following version of the Format_Dates macro formats dates to display with the form mmmm d, yyyy. The following image shows the results of running this macro while a cell with the date June 19, 2014 is active. This date corresponds to the World Cup match between Colombia and Ivory Coast. Date format using VBA:mmmmm-aa The following version of the sample Format_Dates macro causes Excel to display dates in mmmmm-aa format. To see what a date looks like when formatted by this version of the Format_Dates macro, let's go back to the 2014 Brazil World Cup match schedule. The following screenshot shows what the date June 19, 2014 looks like (for the Uruguay-England match) after this macro runs: Date format with VBA: d-mmm-yyyy above, I show versions of the Format_Dates macro that use the format codes d-mmm-aa and dd-mmm-aa. The version of this macro shown in the following image results in a similar date format. The main difference between this version and those shown above is that the next version shows the 4 digits of the year. I run this macro while selecting the cell with the date of the World Cup match between Japan and Greece (June 19, 2014). The resulting date format is shown in the following image: Date format with VBA: dddd, mmmm dd, yyyy The next version of the sample Format_Dates macro causes Excel to display dates in the default long date format in the English (United States) locale. To see what this looks like in practice, check out the image below. This screenshot shows the date of the match between Italy and Costa Rica (June 20, 2014) after the Format_Dates macro has run: So far, this Excel tutorial includes 24 different examples of how you can use Visual Basic for applications to format dates. The date formats entered in the previous sections are relatively simple. These basic date formats include several of the most commonly used date formats in American English. You can also use them as the basis for creating other date formatting macros for less common date formats. However, these basic date formats are not the only ones you can apply. More precisely, once you have a good understanding of how to apply date formats using VBA, you can start creating more complex constructs. To finish this blog post, I introduce one of those date format macros: Date format using VBA: Add a carriage return to dates You can add a carriage return in custom date formats. This allows you to display different elements of a date on different lines within a single cell. Let's see what it looks like in practice: The following screenshot shows an example of a date format with carriage returns. The formatted date is 20 June 2014, corresponding to the World Cup match between Switzerland and France. This example works with a date format that includes only month (using the format code mmmm) and year (using the yyyy format code). You can adjust the I submit below in order to adjust it to your needs and use any other date or format elements. You can use Visual Basic for Applications for these purposes. The next macro (Format_Dates_Carriage_Return) is the one I used to achieve the date format shown in the image above. Some of the elements in this VBA snippet probably look familiar. The following screenshot shows the items I present in the previous sections of this Excel tutorial: There are, however, some other items that I do not present in the previous sections of this blog post. Here are the following 5: #1: With statement, you can generally identify a With statement because of its basic syntax. This syntax is approximately as follows: it starts with a statement of the shape Object. In the case of the Format_Dates_Carriage_Return macro, this opening statement is With Selection. As explained earlier, the Application.Selection property returns the object that is currently selected. When formatting dates using the previous example macro (Format_Dates_Carriage_Return), the selected object is a range of cells. You have 1 or more statements in your body. You can easily identify these 2 statements within the Format_Dates_Carriage_Return because they are indented. Closes with an End With statement. The effect of using a With statement is that all statements within it refer to the same object or structure. In this case: The object referenced by all statements is the object returned by the Selection property. Statements that reference the object returned by Selection are:Statement #1: . NumberFormat mmmm and Chr(10) and yyyy. Statement #2: . RowHeight. RowHeight * 2.#3 instructions: . WrapText: True.I explained each of these instructions below. Using the With statement allows, among others, to simplify macro syntax. I use this statement in other sample macros in Power worksheets, including macros that delete blank rows. Element #2: The Ampersand Operator (&) The first statement within the... Finish with the block is: . NumberFormat mm mm & Chr(10) & yyyy Since, as explained above, this statement works with the object returned by the Application.Selection property, it is the equivalent of: Selection.NumberFormat to mm & Chr(10) & yyyy Most of this statement follows exactly the same structure of (pretty) all other macro examples that I include in the previous sections. There are, however, a couple of new elements. One of those new elements is the ampersand operator (&). Notice how there are 2 ampersands (&) within this statement: within the Visual Basic for Applications environment, the ampersand operator works in a very similar way to how it works in Excel. This means that, within VBA, ampersand (&) is a concatenation operator. In other words, within the macro Format_Dates_Carriage_Return, ampersand (&) concatenates the following 3 expressions: #1: mmmm. #2 expression: expression: Expression #3: yyyy. Expression #2 above takes me to the next and last element that you must consider to understand the first statement within the Con... End with sample macro block: Element of #3: The Chr function The Chr function returns a string. The string that is returned is determined by the given character code that is fed as an argument. In the macro Format_Dates_Carriage_Return, the character code is number 10. This corresponds to a line break character. In other words: Chr(10) is what really adds the return of transport between the month and the year of the date. The second statement within the con... End with block is also new. Let's take a look at the new element it presents: #4: Range.RowHeight Property The Range.RowHeight property allows you to set the height of a row. In the example macro Format_Dates_Carriage_Return, this property is used to duplicate the height of the row for which you are changing the date format. This is done by declaring: . RowHeight. RowHeight * 2 The expression to the right of the equal sign (-) takes the current row height and, using the asterisk operator (*), multiplies it by 2. The result of applying this change of ownership to the sample chart with the 2014 Brazil World Cup match schedule is that a row can now fit the 2 date elements that are separated by the carriage return. Compare the following 2 screenshots to see the difference this statement makes in the date format. The first image shows what happens when the macro is run Format_Dates_Carriage_Return having the statement under analysis. The formatted date, which corresponds to the coincidence between Honduras and Ecuador, is June 20, 2014. The following image shows the result of including the Range.RowHeight property to duplicate the row height. The formatted date corresponds to that of the party between Argentina and Iran (June 21, 2014). Note that this format is not yet what we want. More precisely, the month and year corresponding to the formatted date are displayed on the same line. Element #5, which I explain below, corrects this. If the height of cells whose date format you are modifying is sufficient to fit all elements/lines, you may not need to include this particular statement in the date formatting macro. In other cases, you may need to change the factor by which you multiply the current row height. In other words, instead of using the number 2 at the end of the statement (as I do in the example macro), you might need to use a different number. Using the Range.RowHeight property is optional and does not affect the date format of the selected cells. You can choose to omit it from macros or work with a different property. The reason I use RowHeight in the example it's for illustrative purposes only. In particular, it ensures that the cell I format with this macro displays the full date. Let's take a look at the fifth and of the new elements introduced in the example Format_Dates_Carriage_Return macro: Element #5: Range.WrapText property The Range.WrapText property allows you to determine whether Excel wraps the text within the relevant range object. In the sample Format_Dates_Carriage_Return macro, that relevant range object is the range of cells returned by the Application.Selection property. Within the Format_Dates_Carriage_Return macro, the WrapText property is used to wrap text within its own cell. More precisely, the following statement sets the property to True for all cells within the range returned by the Selection property: . WrapText: True The last image displayed when explaining the Range.RowHeight property above shows the month and year on the same line. The following image allows you to compare the results obtained when I run: (i) the macro version that does not include the WrapText property (for the date of the Argentina-Iran match) and (ii) the macro version that uses the WrapText property (for Germany-Ghana match): Conclusion This Excel tutorial explains the Range.NumberFormat property in great detail and shows how it can be used to format dates with VBA. As you probably will have noticed, correctly applying date formats using VBA usually comes down to knowing and understanding the following 2 topics: item #1: The Range.NumberFormat property. #2 Information: Date Format Codes. In addition to reading about these 2 items, you have seen 25 different date formatting examples with VBA. Such a long list of examples may seem a little excessive, and there are several similarities between some of the date formats I applied. However, these 25 examples are evidence of the flexibility you have when formatting dates with VBA. At the same time, they provide a basis for creating their own macros to apply different date formats. This Excel Vba Date Formatting Tutorial is accompanied by an Excel workbook that contains the data and some versions of the macros I explained earlier. You can get immediate free access to this sample workbook by subscribing to the Power spreadsheet newsletter. Books referenced in this Excel tutorial Alexander, Michael (2015). Excel macros for dummies. Hoboken, NJ: John Wiley & Sons Inc. Walkenbach, John (2013). Excel VBA programming for dummies. Hoboken, NJ: John Wiley & Sons Inc. Inc.