How to Generate a List of Weekend Dates in Excel

Authored by

mohammed looti

January 4, 2026

Introduction to Date Generation Challenges

Generating precise lists of dates is a fundamental requirement in many data analysis and project management tasks within Microsoft Excel. While Excel makes it easy to generate sequential dates, the complexity increases significantly when attempting to filter these sequences based on specific criteria, such as excluding all working days and retaining only the weekend entries. This specific requirement is often necessary for tasks like scheduling maintenance windows, calculating weekend staffing needs, or analyzing time series data that exhibits weekend-specific patterns.

Traditional methods involving simple addition or the standard `WORKDAY` function often fall short because they are primarily designed to bypass weekends, not isolate them. This difficulty necessitates the use of a more specialized and flexible tool within Excel's extensive library of date and time functions. Fortunately, Excel provides a robust solution through the use of the enhanced international workday function, which allows users to define exactly what constitutes a weekend day.

This comprehensive guide details the exact formula required to create a clean, contiguous list of weekend dates only, starting from any given date. We will explore the mechanics of the formula, provide a practical step-by-step example, and dissect the underlying logic of the key function utilized to ensure you can apply this technique confidently in any spreadsheet environment.

The Core Formula for Weekend Date Listing

To efficiently generate a list that exclusively contains weekend dates--specifically, Saturdays and Sundays--you must leverage the powerful WORKDAY.INTL function. This function is an expanded version of the standard `WORKDAY` function, offering crucial flexibility in defining custom work schedules, holidays, and, most importantly for this task, custom weekends.

The specific formula required to advance to the next available weekend date, assuming your starting date is housed in cell **A2**, is structured as follows:

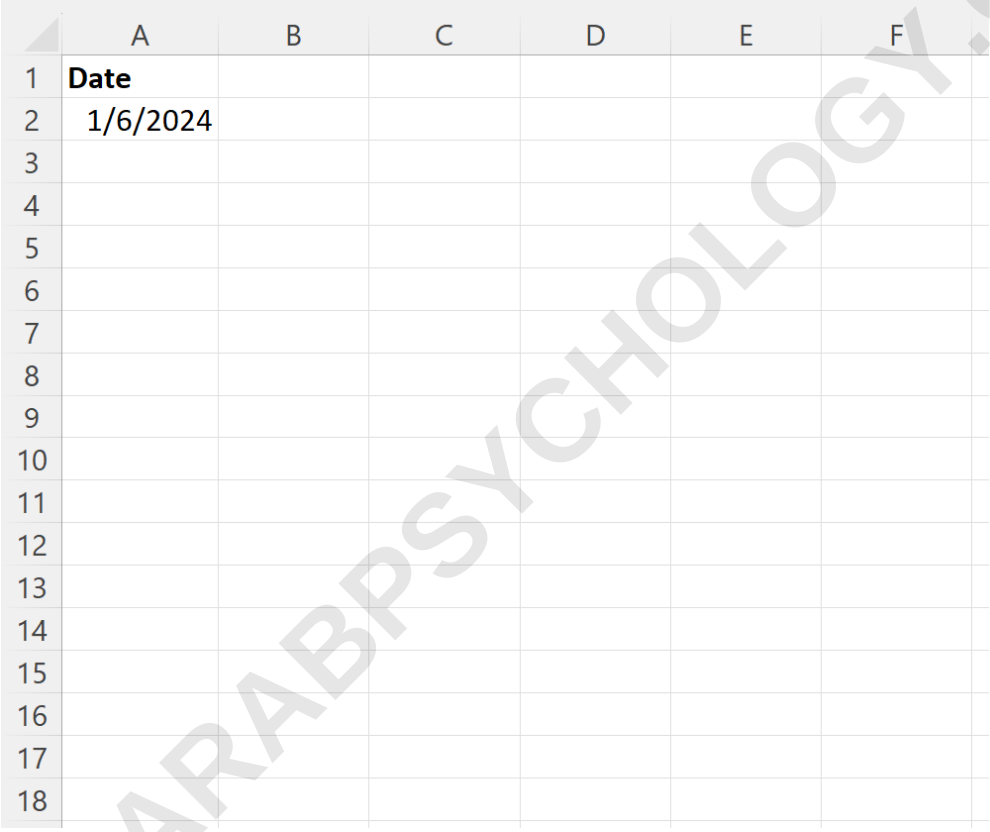
=WORKDAY.INTL(A2,1,"1111100")

This formula operates by taking the initial date in **A2** and calculating the date that is one non-working day ahead. Crucially, the custom weekend argument `"1111100"` redefines what Excel considers a non-working day. This configuration forces the calculation to skip Monday through Friday and land precisely on the next Saturday or Sunday. By setting up this initial calculation in the second cell of your desired column and then extending it down using the fill handle, Excel automatically generates a complete sequence consisting only of weekend days.

Step-by-Step Implementation in Excel

We will now walk through a practical scenario to illustrate the implementation of this technique. Suppose a user needs to generate a sequence of weekend dates beginning on 1/6/2024, which is confirmed to be a Saturday. This process requires three simple steps: defining the start date, entering the formula, and utilizing the fill handle for automatic generation.

Step 1: Define the Start Date. Begin by manually entering the first desired weekend date into cell **A2**. For this example, we input `1/6/2024` into **A2**. It is essential that this starting date is already a weekend day for the sequence to remain clean and accurate, ensuring that the resulting list begins immediately with the first intended entry.

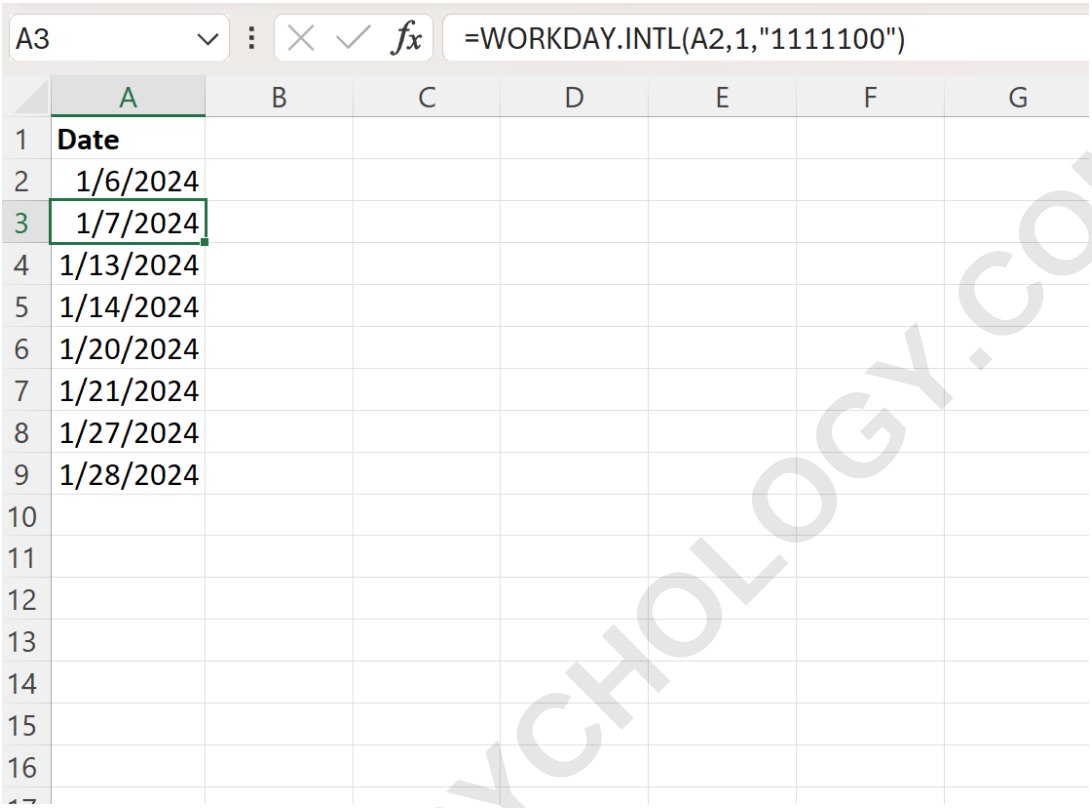


Step 2: Enter the Calculation Formula. Next, move to cell **A3**, the location where the second weekend date will appear. In this cell, we input the core `WORKDAY.INTL` formula, referencing the previous cell (**A2**) as the starting point. This entry tells Excel to calculate the next valid weekend date immediately following 1/6/2024:

=WORKDAY.INTL(A2,1,"1111100")

Step 3: Auto-Generate the List. Once the formula is correctly entered in **A3**, the final step is to

extend this calculation down the column. Click on the fill handle (the small square at the bottom-right corner of cell **A3**) and drag it down to as many cells as required. Excel automatically adjusts the cell references (A2 becomes A3, A3 becomes A4, and so on), generating a contiguous list of subsequent weekend dates.



Upon completion of these steps, column A will contain a fully populated list of dates, each of which falls exclusively on a Saturday or a Sunday, beginning with the specified start date of 1/6/2024. This method provides a dynamic and highly efficient way to manage time-based data filtering using Excel's native date functions.

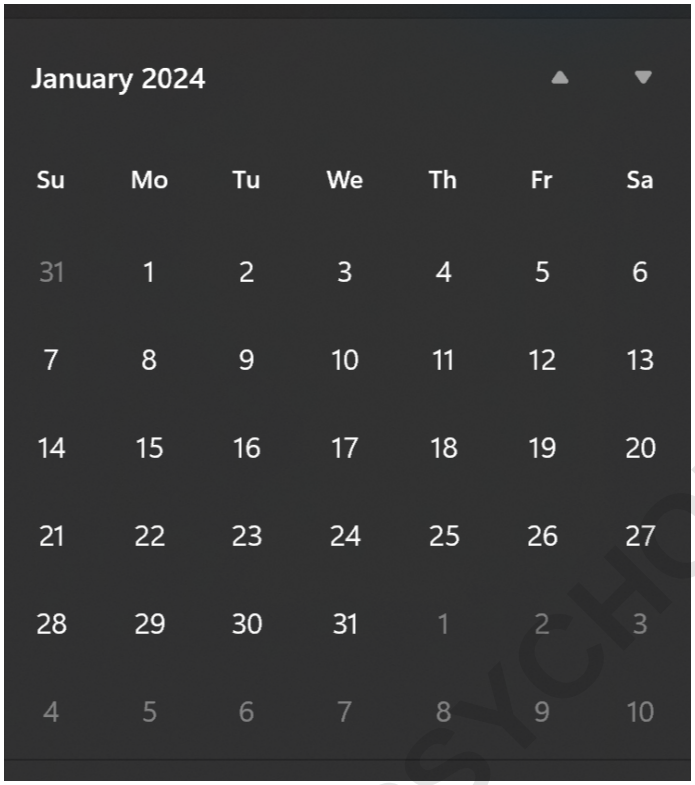
Visualizing the Results and Verification

When working with date functions in Excel, it is always prudent to verify the output, especially when using complex custom parameters like those in `WORKDAY.INTL`. Although Excel displays the output as a standard date (e.g., 1/13/2024), internally, the software stores dates as sequential serial numbers, meaning visual confirmation is necessary to ensure the calculation landed on the correct day of the week, particularly when using unusual workday definitions.

The resulting list, starting at 1/6/2024, should show dates that sequentially jump by one day (Saturday to Sunday) and then by six days (Sunday to the next Saturday). For example, the list should progress: 1/6/2024 (Sat), 1/7/2024 (Sun), 1/13/2024 (Sat), 1/14/2024 (Sun), and so forth,

maintaining the strict weekend boundary.

To definitively verify the accuracy of the generated list, one can compare the calculated dates against a standard calendar for the relevant period. Viewing a calendar confirms that every date generated in column A--from 1/6/2024 onward--occurs precisely on a weekend day (Saturday or Sunday), validating the function's ability to bypass the five business days effectively.



As confirmed by the calendar visualization, each date derived from the formula adheres strictly to the requirement of being a Saturday or Sunday. If any date in the sequence were to fall on a Monday through Friday, it would immediately indicate an error in the custom argument pattern provided to the function, requiring immediate adjustment of the binary string.

Deep Dive into the WORKDAY.INTL Function

Understanding the structure of the `WORKDAY.INTL` function is key to mastering this technique and adapting it for different scheduling needs. Unlike the simpler `WORKDAY` function, which is limited to defining weekends as only Saturday and Sunday, `WORKDAY.INTL` provides an optional argument that grants complete control over the weekend definition. This advanced capability is what makes the listing of weekends possible.

The fundamental syntax for the function is structured as follows:

WORKDAY.INTL(start_date, days, ,)

Our specific formula uses the first three arguments to achieve the desired outcome. Let us break down the purpose of each critical parameter utilized in our application:

start_date: This is the required initial date from which the calculation begins (e.g., the value in cell **A2**). It serves as the reference point for counting forward or backward in time, depending on the sign of the `days` argument.
days: This parameter specifies the number of non-working days you wish to move forward or backward from the **start_date**. A value of `1` instructs the function to find the very next non-working day, relative to the starting point.
weekend: This is the crucial, optional parameter that defines which days are considered workdays (0) and which are non-workdays (1). It can be supplied either as a numerical code (1 through 17) for standard patterns or, as we used, a custom seven-digit binary string.

For more complex scheduling, an optional fourth argument, , can also be supplied as a range of dates to exclude specific statutory or company holidays from the resulting list, further enhancing the function's utility in professional environments.

Understanding the Custom Weekend Argument ("1111100")

The core innovation behind creating a weekend-only list lies entirely in the custom **weekend** argument: `"1111100"`. This seven-character string acts as a powerful binary mask, allowing the user to precisely specify the work status of each day of the week, beginning its sequence count with Monday.

The position of the digits corresponds directly to the days of the week in the following order: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday. Within this string, the interpretation of the digits is critical:

The digit **1** signifies a **non-working day** (a day that should be skipped when counting working days).
The digit **0** signifies a **working day** (a day that must be included when counting working days).

By applying `"1111100"`, we effectively tell Excel that Monday through Friday are non-working days (1s), and conversely, Saturday and Sunday are the only working days (0s). The key to the formula's success is that we have essentially reversed the conventional definition of the work week.

When the function is executed using `=WORKDAY.INTL(A2, 1, "1111100")`, Excel performs this unique calculation: it takes the date in **A2** and looks for the date that is one **working day** (a day

marked as 0) away. Since the only days marked as working days (0s) are Saturday and Sunday, the function is forced to skip over Monday through Friday, landing only on the next sequential weekend day. This intentional misapplication of the function allows us to isolate the desired dates with remarkable precision and efficiency.

Alternative Uses and Customization

The power of the WORKDAY.INTL function lies in its profound adaptability. While our primary example focused on the standard Saturday/Sunday weekend pattern prevalent in Western countries, the custom argument allows for straightforward modification to accommodate any regional or organizational work schedule, such as those that observe Friday and Saturday as the weekend.

To adjust the formula for alternative weekend structures, simply modify the seven-digit binary string to reflect the desired pattern. For instance, if the required sequence consists of Friday and Saturday only, the string must be altered to assign a '0' to the Friday and Saturday positions while assigning a '1' to all other positions (Monday, Tuesday, Wednesday, Thursday, and Sunday).

The revised custom string for generating a Friday/Saturday list would therefore be: `"1111001"`. Here, Friday and Saturday are marked as '0' (working days for the purpose of the function) and Sunday is marked as '1' (non-working day). If this new string were used in the formula, Excel would generate a sequential list consisting only of Friday and Saturday dates, demonstrating the function's versatility in scheduling contexts across the globe. This level of customization ensures that the technique is universally applicable regardless of the local definition of the non-working period.

Summary of Key Concepts and Further Resources

The successful generation of a weekend-only date list in Excel relies entirely on the precise definition provided by the `WORKDAY.INTL` function's custom weekend parameter. By utilizing the binary string `"1111100"`, we strategically reclassify the five typical business days as days to be skipped, thus isolating the Saturdays and Sundays as the only calculable steps in the sequence. This approach is highly reliable and easy to replicate.

This method provides a robust and dynamic alternative to complex array formulas or iterative calculations often required for filtering date sequences. Mastering the use of the custom string mask opens up possibilities for generating lists based on any defined weekly pattern--for example, listing only Tuesdays and Thursdays, or only Mondays, simply by adjusting the placement of the '0's within the seven-digit string.

For users seeking to expand their knowledge of date and time manipulation in Excel, the following

related functions are essential tools that complement the techniques discussed here. These functions are often used in conjunction with `WORKDAY.INTL` for comprehensive schedule planning and analysis:

`NETWORKDAYS.INTL`: Used to calculate the number of workdays between two dates, utilizing the same custom weekend string flexibility.

`EDATE`: Used to calculate the date a specified number of months before or after a start date, maintaining the day of the month.

`EOMONTH`: Returns the serial number for the last day of the month before or after a specified number of months.

The complete official documentation for the `WORKDAY.INTL` function, including all standard numerical codes for the weekend argument, can be found on the Microsoft Support website, providing additional technical depth for advanced users who require further customization.