Python (programming language)

Python

Paradigm	Multi-paradigm: object-oriented,[1] procedural (imperative), functional, structured, reflective
Designed by	Guido van Rossum
Developer	Python Software Foundation
First appeared	20 February 1991[2]
Stable release	3.13.2 / 4 February 2025
Typing discipline	duck, dynamic, strong;[3] optional type annotations (since 3.5, but those hints are ignored, except with unofficial tools)[4]
OS	**Tier 1**: 64-bit Linux, macOS; 64- and 32-bit Windows 10+[5] **Tier 2**: E.g. 32-bit WebAssembly (WASI) **Tier 3**: 64-bit Android,[6] iOS, FreeBSD, and (32-bit) Raspberry Pi OS Unofficial (or has been known to work): Other Unix-like/BSD variants) and a few other platforms[7][8][9]
License	Python Software Foundation License
Filename extensions	.py, .pyw, .pyz,[10] .pyi, .pyc, .pyd
Website	python.org (https://www.python.org/)
Major implementations	
CPython, PyPy, Stackless Python, MicroPython, CircuitPython, IronPython, Jython	
Dialects	
Cython, RPython, Starlark[11]	
Influenced by	
ABC,[12] Ada,[13] ALGOL 68,[14] APL,[15] C,[16] C++,[17] CLU,[18] Dylan,[19] Haskell,[20][15] Icon,[21] Lisp,[22] Modula-3,[14][17] Perl,[23] Standard ML[15]	
Influenced	

Apache Groovy, Boo, Cobra, CoffeeScript,[24] D, F#, GDScript, Go, JavaScript,[25][26] Julia,[27] Mojo,[28] Nim, Ring,[29] Ruby,[30] Swift,[31] V[32]

Python Programming at Wikibooks

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.[33]

Python is dynamically type-checked and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.[34][35]

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0.[36] Python 2.0 was released in 2000. Python 3.0, released in 2008, was a major revision not completely backward-compatible with earlier versions. Python 2.7.18, released in 2020, was the last release of Python 2.[37]

Python consistently ranks as one of the most popular programming languages, and has gained widespread use in the machine learning community.[38][39][40][41]

History

Python was conceived in the late 1980s[42] by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC programming language, which was inspired by SETL,[43] capable of exception handling and interfacing with the Amoeba operating system.[12] Its implementation began in December 1989.[44] Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's "benevolent dictator for life" (BDFL), a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker[45] (he has since come out of retirement and is self-titled "BDFL-emeritus"). In January 2019, active Python core developers elected a five-member Steering Council to lead the project.[46][47]

The designer of Python, Guido van Rossum, at PyCon US 2024

The name Python is said to come from the British comedy series Monty Python's Flying Circus.[48]

Python 2.0 was released on 16 October 2000, with many major new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support.[49] Python 2.7's end-of-life was initially set for 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3.[50][51] It no longer receives security patches or updates.[52][53] While Python

2.7 and older versions are officially unsupported, a different unofficial Python implementation, PyPy, continues to support Python 2, i.e. "2.7.18+" (plus 3.10), with the plus meaning (at least some) "backported security updates".[54]

Python 3.0 was released on 3 December 2008, with some new semantics and changed syntax. At least every Python release since (now unsupported) 3.5 has added some syntax to the language, and a few later releases have dropped outdated modules, and changed semantics, at least in a minor way.

As of 12 March 2025, Python 3.13 is the latest stable release. Python 3.13 currently receives full bug-fix and security updates, while Python 3.12—released in October 2023—will have active bug-fix support only until April 2025. Python 3.9[55] is the oldest supported version of Python (albeit in the 'security support' phase), due to Python 3.8 reaching end-of-life.[56][57] Starting with 3.13, it and later versions have 2 years of full support (up from one and a half), followed by 3 years of security support (for same total support as before).

Security updates were expedited in 2021 (and again twice in 2022, and more fixed in 2023 and in September 2024 for Python 3.12.6 down to 3.8.20), since all Python versions were insecure (including 2.7[58]) because of security issues leading to possible remote code execution[59] and web-cache poisoning.[60]

Python 3.10 added the `|` union type operator[61] and the `match` and `case` keywords (for structural pattern matching statements). 3.11 expanded exception handling functionality. Python 3.12 added the new keyword `type`. Notable changes in 3.11 from 3.10 include increased program execution speed and improved error reporting.[62] Python 3.11 claims to be between 10 and 60% faster than Python 3.10, and Python 3.12 adds another 5% on top of that. It also has improved error messages (again improved in 3.14), and many other changes.

Python 3.13 introduces more syntax for types, a new and improved interactive interpreter (REPL), featuring multi-line editing and color support; an incremental garbage collector (producing shorter pauses for collection in programs with a lot of objects, and addition to the improved speed in 3.11 and 3.12), and an *experimental* just-in-time (JIT) compiler (such features, can/needs to be enabled specifically for the increase in speed),[63] and an *experimental* free-threaded build mode, which disables the global interpreter lock (GIL), allowing threads to run more concurrently, that latter feature enabled with `python3.13t` or `python3.13t.exe`.

Python 3.13 introduces some change in behavior, i.e. new "well-defined semantics", fixing bugs (plus many removals of deprecated classes, functions and methods, and removed some of the C API and outdated modules): "The [old] implementation of `locals()` and `frame.f_locals` is slow, inconsistent and buggy [and it] has many corner cases and oddities. Code that works around those may need to be changed. Code that uses `locals()` for simple templating, or print debugging, will continue to work correctly."[64]

Python 3.13 introduces some changes in behavior, including new "well-defined semantics" and the removal of deprecated classes, functions, methods, and outdated modules from the C API. One of the most significant additions is the experimental free-threaded build mode, which disables the Global Interpreter Lock (GIL), a feature of CPython that previously prevented multiple threads from executing Python bytecode simultaneously. This optional build, introduced through PEP 703, enables better exploitation of multi-core CPUs. By allowing multiple threads to

run Python code in parallel, the free-threaded mode addresses long-standing performance bottlenecks associated with the GIL, offering a new path for parallelism in Python without resorting to multiprocessing or external concurrency frameworks.[65]
Some (more) standard library modules and many deprecated classes, functions and methods, will be removed in Python 3.15 or 3.16.[66][67]
Python 3.11 adds Sigstore digital verification signatures for all CPython artifacts (in addition to PGP). Since use of PGP has been criticized by security practitioners, Python is moving to Sigstore exclusively and dropping PGP from 3.14.[68]
Python 3.14 is now in alpha 3; regarding possible change to annotations: "In Python 3.14, `from __future__ import` annotations will continue to work as it did before, converting annotations into strings."[69]
PEP 711 proposes PyBI: a standard format for distributing Python Binaries.[70]
Python 3.15 will "Make UTF-8 mode default",[71] the mode exists in all current Python versions, but currently needs to be opted into. UTF-8 is already used, by default, on Windows (and elsewhere), for most things, but e.g. to `open` files it's not and enabling also makes code fully cross-platform, i.e. use UTF-8 for everything on all platforms.

Design philosophy and features

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of their features support functional programming and aspect-oriented programming (including metaprogramming[72] and metaobjects).[73] Many other paradigms are supported via extensions, including design by contract[74][75] and logic programming.[76] Python is often referred to as a *'glue language'*[77] because it can seamlessly integrate components written in other languages.
Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management.[78] It uses dynamic name resolution (late binding), which binds method and variable names during program execution.
Its design offers some support for functional programming in the Lisp tradition. It has `filter`,`map`and`reduce` functions; list comprehensions, dictionaries, sets, and generator expressions.[79] The standard library has two modules (`itertools` and `functools`) that implement functional tools borrowed from Haskell and Standard ML.[80]
Its core philosophy is summarized in the Zen of Python (PEP 20), which includes aphorisms such as:[81]

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

However, Python features regularly violate these principles and have received criticism for adding unnecessary language bloat.[82] Responses to these criticisms are that the Zen of Python is a guideline rather than a rule.[83] The addition of some new features had been so controversial that Guido van Rossum resigned as Benevolent Dictator for Life following vitriol over the addition of the assignment expression operator in Python 3.8.[84][85]

Nevertheless, rather than building all of its functionality into its core, Python was designed to be highly extensible via modules. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.[42]

Python claims to strive for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one—and preferably only one—obvious way to do it." philosophy.[81] In practice, however, Python provides many ways to achieve the same task. There are, for example, at least three ways to format a string literal, with no certainty as to which one a programmer should use.[86] Alex Martelli, a Fellow at the Python Software Foundation and Python book author, wrote: "To describe something as 'clever' is *not* considered a compliment in the Python culture."[87]

Python's developers usually strive to avoid premature optimization and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity.[88] Execution speed can be improved by moving speed-critical functions to extension modules written in languages such as C, or by using a just-in-time compiler like PyPy. It is also possible to cross-compile to other languages, but it either doesn't provide the full speed-up that might be expected, since Python is a very dynamic language, or a restricted subset of Python is compiled, and possibly semantics are slightly changed.[89]

Python's developers aim for it to be fun to use. This is reflected in its name—a tribute to the British comedy group Monty Python[90]—and in occasionally playful approaches to tutorials and reference materials, such as the use of the terms "spam" and "eggs" (a reference to a Monty Python sketch) in examples, instead of the often-used "foo" and "bar".[91][92] A common neologism in the Python community is *pythonic*, which has a wide range of meanings related to program style. "Pythonic" code may use Python idioms well, be natural or show fluency in the language, or conform with Python's minimalist philosophy and emphasis on readability.[93]

An example of Python code and indentation

Syntax and semantics

Python is meant to be an easily readable language. Its formatting is visually uncluttered and often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are allowed but rarely used. It has fewer syntactic exceptions and special cases than C or Pascal.[94]

```python
class CodeExample():
    def printStatement(self):
        print('Hello World!')

def main():
    classEx = CodeExample()
    classEx.printStatement()
if __name__ == "__main__":
    main()
```

An example of Python code and indentation

Indentation

Python uses whitespace indentation, rather than curly brackets or keywords, to delimit blocks. An increase in indentation comes after certain statements; a decrease in indentation signifies the end of the current block.[95] Thus, the program's visual structure accurately represents its semantic structure.[96] This feature is sometimes termed the off-side rule. Some other languages use indentation this way; but in most, indentation has no semantic meaning. The recommended indent size is four spaces.[97]

```csharp
using System;

// Voorbeeld van 'n Hallo Wêreld program
namespace HalloWêreld
{
   class Program
   {
      static void Main(string[] args)
      {
         Console.WriteLine("Hallo Wêreld!");
      }
   }
}
```

Example of C# code with curly braces and semicolons

Statements and control flow

Python's statements include:

- The assignment statement, using a single equals sign =
- The `if` statement, which conditionally executes a block of code, along with `else` and `elif` (a contraction of `else if`)
- The `for` statement, which iterates over an *iterable* object, capturing each element to a local variable for use by the attached block
- The `while` statement, which executes a block of code as long as its condition is true
- The `try` statement, which allows exceptions raised in its attached code block to be caught and handled by `except` clauses (or new syntax `except*` in Python 3.11 for exception groups[98]); it also ensures that clean-up code in a `finally` block is always run regardless of how the block exits

- The `raise` statement, used to raise a specified exception or re-raise a caught exception
- The `class` statement, which executes a block of code and attaches its local namespace to a class, for use in object-oriented programming
- The `def` statement, which defines a function or method
- The `with` statement, which encloses a code block within a context manager (for example, acquiring a lock before it is run, then releasing the lock; or opening and closing a file), allowing resource-acquisition-is-initialization (RAII)-like behavior and replacing a common try/finally idiom[99]
- The `break` statement, which exits a loop
- The `continue` statement, which skips the rest of the current iteration and continues with the next
- The `del` statement, which removes a variable—deleting the reference from the name to the value, and producing an error if the variable is referred to before it is redefined
- The `pass` statement, serving as a NOP, syntactically needed to create an empty code block
- The `assert` statement, used in debugging to check for conditions that should apply
- The `yield` statement, which returns a value from a generator function (and also an operator); used to implement coroutines
- The `return` statement, used to return a value from a function
- The `import` and `from` statements, used to import modules whose functions or variables can be used in the current program
- The `match` and `case` statements, an analog of the switch statement construct, that compares an expression against one or more cases as a control-of-flow measure.

The assignment statement (`=`) binds a name as a reference to a separate, dynamically allocated object. Variables may subsequently be rebound at any time to any object. In Python, a variable name is a generic reference holder without a fixed data type; however, it always refers to *some* object with a type. This is called dynamic typing—in contrast to statically-typed languages, where each variable may contain only a value of a certain type.

Python does not support tail call optimization or first-class continuations, and, according to Van Rossum, it never will.[100][101] However, better support for coroutine-like functionality is provided by extending Python's generators.[102] Before 2.5, generators were lazy iterators; data was passed unidirectionally out of the generator. From Python 2.5 on, it is possible to pass data back into a generator function; and from version 3.3, it can be passed through multiple stack levels.[103]

Expressions

Python's expressions include:

- The `+`, `-`, and `*` operators for mathematical addition, subtraction, and multiplication are similar to other languages, but the behavior of division differs. There are two types of divisions in Python: floor division (or integer division) `//` and floating-point `/` division.[104] Python uses the `**` operator for exponentiation.
- Python uses the `+` operator for string concatenation. Python uses the `*` operator for duplicating a string a specified number of times.
- The `@` infix operator is intended to be used by libraries such as NumPy for matrix multiplication.[105][106]
- The syntax `:=`, called the "walrus operator", was introduced in Python 3.8. It assigns values to variables as part of a larger expression.[107]
- In Python, `==` compares by value. Python's `is` operator may be used to compare object identities (comparison by reference), and comparisons may be chained—for example, `a <= b <= c`.
- Python uses `and`, `or`, and `not` as Boolean operators.
- Python has a type of expression named a *list comprehension*, and a more general expression named a *generator expression*.[79]
- Anonymous functions are implemented using lambda expressions; however, there may be only one expression in each body.
- Conditional expressions are written as `x if c else y`[108] (different in order of operands from the `c ? x : y` operator common to many other languages).
- Python makes a distinction between lists and tuples. Lists are written as `[1, 2, 3]`, are mutable, and cannot be used as the keys of dictionaries (dictionary keys must be immutable in Python). Tuples, written as `(1, 2, 3)`, are immutable and thus can be used as keys of dictionaries, provided all of the tuple's elements are immutable. The `+` operator can be used to concatenate two tuples, which does not directly modify their contents, but produces a new tuple containing the elements of both. Thus, given the variable `t` initially equal to `(1, 2, 3)`, executing `t = t + (4, 5)` first evaluates `t + (4, 5)`, which yields `(1, 2, 3, 4, 5)`, which is then assigned back to `t`—thereby effectively "modifying the contents" of `t` while conforming to the immutable nature of tuple objects. Parentheses are optional for tuples in unambiguous contexts.[109]
- Python features *sequence unpacking* where multiple expressions, each evaluating to anything that can be assigned (to a variable, writable property, etc.) are associated in an identical manner to that forming tuple literals—and, as a whole, are put on the left-hand side of the equal sign in an assignment statement. The statement expects an *iterable* object on the right-hand side of the equal sign that produces the same number of values as the provided

writable expressions; when iterated through them, it assigns each of the produced values to the corresponding expression on the left.[110]

- Python has a "string format" operator % that functions analogously to `printf` format strings in C—e.g. `"spam=%s eggs=%d" % ("blah", 2)` evaluates to `"spam=blah eggs=2"`. In Python 2.6+ and 3+, this was supplemented by the `format()` method of the `str` class, e.g. `"spam={0} eggs={1}".format("blah", 2)`. Python 3.6 added "f-strings": `spam = "blah"; eggs = 2; f'spam={spam} eggs={eggs}'`.[111]
- Strings in Python can be concatenated by "adding" them (with the same operator as for adding integers and floats), e.g. `"spam" + "eggs"` returns `"spameggs"`. If strings contain numbers, they are added as strings rather than integers, e.g. `"2" + "2"` returns `"22"`.
- Python has various string literals:
  - Delimited by single or double quotes; unlike in Unix shells, Perl, and Perl-influenced languages, single and double quotes work the same. Both use the backslash (`\`) as an escape character. String interpolation became available in Python 3.6 as "formatted string literals".[111]
  - Triple-quoted (beginning and ending with three single or double quotes), which may span multiple lines and function like here documents in shells, Perl, and Ruby.
  - Raw string varieties, denoted by prefixing the string literal with `r`. Escape sequences are not interpreted; hence raw strings are useful where literal backslashes are common, such as regular expressions and Windows-style paths. (Compare "@-quoting" in C#.)
- Python has array index and array slicing expressions in lists, denoted as `a[key]`, `a[start:stop]` or `a[start:stop:step]`. Indexes are zero-based, and negative indexes are relative to the end. Slices take elements from the *start* index up to, but not including, the *stop* index. The third slice parameter, called *step* or *stride*, allows elements to be skipped and reversed. Slice indexes may be omitted—for example, `a[:]` returns a copy of the entire list. Each element of a slice is a shallow copy.

In Python, a distinction between expressions and statements is rigidly enforced, in contrast to languages such as Common Lisp, Scheme, or Ruby. This leads to duplicating some functionality. For example:

- List comprehensions vs. `for`-loops
- Conditional expressions vs. `if` blocks
- The `eval()` vs. `exec()` built-in functions (in Python 2, `exec` is a statement); the former is for expressions, the latter is for statements

Statements cannot be a part of an expression—so list and other comprehensions or lambda expressions, all being expressions, cannot contain statements. A particular case is that an assignment statement such as `a = 1` cannot form part of the conditional expression of a conditional statement.

Methods

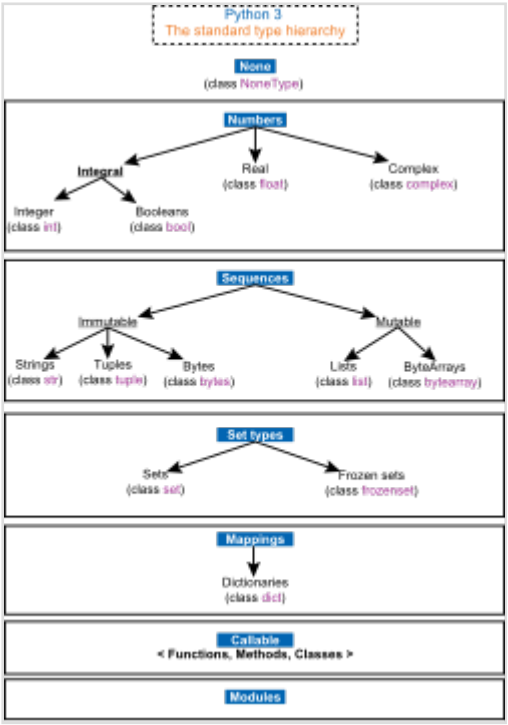
Methods on objects are functions attached to the object's class; the syntax `instance.method(argument)` is, for normal methods and functions, syntactic sugar for `Class.method(instance, argument)`. Python methods have an explicit `self` parameter to access instance data, in contrast to the implicit self (or `this`) in some other object-oriented programming languages (e.g., C++, Java, Objective-C, Ruby).[112] Python also provides methods, often called *dunder methods* (due to their names beginning and ending with double-underscores), to allow user-defined classes to modify how they are handled by native operations including length, comparison, in arithmetic operations and type conversion.[113]

Typing

Python uses duck typing and has typed objects but untyped variable names. Type constraints are not checked at compile time; rather, operations on an object may fail, signifying that it is not of a suitable type. Despite being dynamically typed, Python is strongly typed, forbidding operations that are not well-defined (for example, adding a number to a string) rather than silently attempting to make sense of them.

Python allows programmers to define their own types using classes, most often used for object-oriented programming. New instances of classes are constructed by calling the class (for example, `SpamClass()` or `EggsClass()`), and the classes are instances of the metaclass `type` (itself an instance of itself), allowing metaprogramming and reflection.

Before version 3.0, Python had two kinds of classes (both using the same syntax): *old-style* and *new-style*;[114] current Python versions only support the semantics of the new style.



The standard type hierarchy in Python 3

Python supports optional type annotations.[4][115] These annotations are not enforced by the language, but may be used by external tools such as **mypy** to catch errors.[116][117] Mypy also supports a Python compiler called mypyc, which leverages type annotations for optimization.[118]

Summary of Python 3's built-in types

Type	Mutability	Description	Syntax examples
`bool`	immutable	Boolean value	`True` `False`

`bytearray`	mutable	Sequence of bytes	`bytearray(b'Some ASCII')`
			`bytearray(b"Some ASCII")`
			`bytearray([119, 105, 107, 105])`

`bytes`	immutable	Sequence of bytes	`b'Some ASCII'`
			`b"Some ASCII"`
			`bytes([119, 105, 107, 105])`

`complex`	immutable	Complex number with real and imaginary parts	`3+2.7j` `3 + 2.7j`
`dict`	mutable	Associative array (or dictionary) of key and value pairs; can contain mixed types (keys and values), keys must be a hashable type	`{'key1': 1.0, 3: False}` `{}`
`types.EllipsisType`	immutable	An ellipsis placeholder to be used as an index in NumPy arrays	`...` `Ellipsis`
`float`	immutable	Double-precision floating-point number. The precision is machine-dependent but in practice is generally implemented as a 64-bit IEEE 754 number with 53 bits of precision.[119]	`1.33333`
`frozenset`	immutable	Unordered set, contains no duplicates; can	`frozenset([4.0, 'string', True])`

		contain mixed types, if hashable	
`int`	immutable	Integer of unlimited magnitude[120]	`42`
`list`	mutable	List, can contain mixed types	`[4.0, 'string', True]` `[]`
`types.NoneType`	immutable	An object representing the absence of a value, often called null in other languages	`None`
`types.NotImplementedType`	immutable	A placeholder that can be returned from overloaded operators to indicate unsupported operand types.	`NotImplemented`
`range`	immutable	An *immutable sequence* of numbers commonly used for looping a specific number of times in `for` loops[121]	`range(-1, 10)` `range(10, -5, -2)`
`set`	mutable	Unordered set, contains no duplicates; can contain mixed types, if hashable	`{4.0, 'string', True}` `set()`

`str`	immutable	A character string: sequence of Unicode codepoints	`'Wikipedia'`
			`"Wikipedia"`
			`"""Spanning`
			`multiple`
			`lines"""`

			`Spanning multiple lines`

`tuple`	immutable	Can contain mixed types	`(4.0, 'string', True)`
			`('single element',)`
			`()`

Arithmetic operations

Python has the usual symbols for arithmetic operators (`+`, `-`, `*`, `/`), the floor division operator `//` and the modulo operation `%` (where the remainder can be negative, e.g. `4 % -3 == -2`). It also has `**` for exponentiation, e.g. `5**3 == 125` and `9**0.5 == 3.0`, and a matrix-multiplication operator `@` .[122] These operators work like in traditional math; with the same precedence rules, the operators infix (`+` and `-` can also be unary to represent positive and negative numbers respectively).

The division between integers produces floating-point results. The behavior of division has changed significantly over time:[123]

- Current Python (i.e. since 3.0) changed `/` to always be floating-point division, e.g. `5/2 == 2.5`.
- The floor division `//` operator was introduced. So `7//3 == 2`, `-7//3 == -3`, `7.5//3 == 2.0` and `-7.5//3 == -3.0`. Adding `from __future__ import division` causes a module used in Python 2.7 to use Python 3.0 rules for division (see above).

In Python terms, `/` is *true division* (or simply *division*), and `//` is *floor division*. `/` before version 3.0 is *classic division*.[123]

Rounding towards negative infinity, though different from most languages, adds consistency. For instance, it means that the equation `(a + b)//b == a//b + 1` is always true. It also means that the equation `b*(a//b) + a%b == a` is valid for both positive and negative values of `a`. However, maintaining the validity of this equation means that while the result of `a%b` is, as expected, in the half-open interval [0, *b*), where `b` is a positive integer, it has to lie in the interval (*b*, 0] when `b` is negative.[124]

Python provides a `round` function for rounding a float to the nearest integer. For tie-breaking, Python 3 uses round to even: `round(1.5)` and `round(2.5)` both produce `2`.[125] Versions before 3 used round-away-from-zero: `round(0.5)` is `1.0`, `round(-0.5)` is `-1.0`.[126]

Python allows Boolean expressions with multiple equality relations in a manner that is consistent with general use in mathematics. For example, the expression `a < b < c` tests whether `a` is less than `b` and `b` is less than `c`.[127] C-derived languages interpret this expression differently: in C, the expression would first evaluate `a < b`, resulting in 0 or 1, and that result would then be compared with `c`.[128]

Python uses arbitrary-precision arithmetic for all integer operations. The `Decimal` type/class in the `decimal` module provides decimal floating-point numbers to a pre-defined arbitrary precision and several rounding modes.[129] The `Fraction` class in the `fractions` module provides arbitrary precision for rational numbers.[130]
Due to Python's extensive mathematics library and the third-party library NumPy, it is frequently used as a scientific scripting language to aid in problems such as numerical data processing and manipulation.[131][132]

Function syntax

Functions are created in Python using the `def` keyword. In Python, you define the function as if you were calling it, by typing the function name and then the attributes required. Here is an example of a function that will print whatever is given:

```
def printer(input1, input2="already there"):
    print(input1)
    print(input2)

printer("hello")

# Example output:
# hello
# already there
```

If you want the attribute to have a set value if no value is given, use the variable-defining syntax inside the function definition.

Programming examples

"Hello, World!" program:

```
print('Hello, world!')
```

Program to calculate the factorial of a positive integer:

```
n = int(input('Type a number, and its factorial will be printed: '))

if n < 0:
    raise ValueError('You must enter a non-negative integer')

factorial = 1
for i in range(2, n + 1):
    factorial *= i

print(factorial)
```

Libraries

Python's large standard library[133] is commonly cited as one of its greatest strengths. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary-precision decimals,[129] manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications—for example, the Web Server Gateway Interface (WSGI) implementation `wsgiref` follows PEP 333[134]—but most are specified by their code, internal documentation, and test suites. However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of 13 March 2025, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 614,339[135] packages with a wide range of functionality, including:

- Automation
- Data analytics
- Databases
- Documentation
- Graphical user interfaces
- Image processing
- Machine learning
- Mobile apps
- Multimedia
- Computer networking
- Scientific computing
- System administration
- Test frameworks
- Text processing
- Web frameworks
- Web scraping

Development environments

Most Python implementations (including CPython) include a read–eval–print loop (REPL), permitting them to function as a command line interpreter for which users enter statements sequentially and receive results immediately.

Python also comes with an Integrated development environment (IDE) called IDLE, which is more beginner-oriented.

Other shells, including IDLE and IPython, add further abilities such as improved auto-completion, session state retention, and syntax highlighting.
As well as standard desktop integrated development environments including PyCharm, IntelliJ Idea, Visual Studio Code etc, there are web browser-based IDEs, including SageMath, for developing science- and math-related programs; PythonAnywhere, a browser-based IDE and hosting environment; and Canopy IDE, a commercial IDE emphasizing scientific computing.[136]

Implementations

Reference implementation

CPython is the reference implementation of Python. It is written in C, meeting the C89 standard (Python 3.11 uses C11[137]) with several select C99 features. CPython includes its own C extensions, but third-party extensions are not limited to older C versions—e.g. they can be implemented with C11 or C++.[138][139] CPython compiles Python programs into an intermediate bytecode[140] which is then executed by its virtual machine.[141] CPython is distributed with a large standard library written in a mixture of C and native Python, and is available for many platforms, including Windows (starting with Python 3.9, the Python installer deliberately fails to install on Windows 7 and 8;[142][143] Windows XP was supported until Python 3.5) and most modern Unix-like systems, including macOS (and Apple M1 Macs, since Python 3.9.1, with experimental installer), with unofficial support for VMS.[144] Platform portability was one of its earliest priorities.[145] (During Python 1 and 2 development, even OS/2 and Solaris were supported,[146] but support has since been dropped for many platforms.)
All current Python versions (i.e. since 3.7) only support operating systems with multi-threading support.

Other implementations

All alternative implementations have at least slightly different semantics (e.g. may have unordered dictionaries, unlike all current Python versions), e.g. with the larger Python ecosystem, such as with supporting the C Python API of with PyPy:

- PyPy is a fast, compliant interpreter of Python 2.7 and 3.10.[147][148] Its just-in-time compiler often brings a significant speed improvement over CPython, but some libraries written in C cannot be used with it.[149] It has e.g. RISC-V support.
- Codon is a language with an ahead-of-time (AOT) compiler, that (AOT) compiles a statically-typed Python-like language with "syntax and semantics are nearly identical to Python's, there are some notable differences"[150] e.g. it uses 64-bit machine integers, for speed, not arbitrary like Python, and it claims speedups over CPython are usually on the order of 10–100x. It compiles to machine code (via LLVM) and supports native multithreading.[151] Codon can also compile to Python extension modules that can be imported and used from Python.

- Stackless Python is a significant fork of CPython that implements microthreads; it does not use the call stack in the same way, thus allowing massively concurrent programs. PyPy also has a stackless version.[152]
- MicroPython and CircuitPython are Python 3 variants optimized for microcontrollers, including Lego Mindstorms EV3.[153]
- Pyston is a variant of the Python runtime that uses just-in-time compilation to speed up the execution of Python programs.[154]
- Cinder is a performance-oriented fork of CPython 3.8 that contains a number of optimizations, including bytecode inline caching, eager evaluation of coroutines, a method-at-a-time JIT, and an experimental bytecode compiler.[155]
- Snek[156][157][158] Embedded Computing Language (compatible with e.g. 8-bit AVR microcontrollers such as ATmega 328P-based Arduino, as well as larger ones compatible with MicroPython) "is Python-inspired, but it is not Python. It is possible to write Snek programs that run under a full Python system, but most Python programs will not run under Snek."[159] It is an imperative language not including OOP / classes, unlike Python, and simplifying to one number type with 32-bit single-precision (similar to JavaScript, except smaller).

No longer supported implementations

Other just-in-time Python compilers have been developed, but are now unsupported:

- Google began a project named Unladen Swallow in 2009, with the aim of speeding up the Python interpreter five-fold by using the LLVM, and of improving its multithreading ability to scale to thousands of cores,[160] while ordinary implementations suffer from the global interpreter lock.
- Psyco is a discontinued just-in-time specializing compiler that integrates with CPython and transforms bytecode to machine code at runtime. The emitted code is specialized for certain data types and is faster than the standard Python code. Psyco does not support Python 2.7 or later.
- PyS60 was a Python 2 interpreter for Series 60 mobile phones released by Nokia in 2005. It implemented many of the modules from the standard library and some additional modules for integrating with the Symbian operating system. The Nokia N900 also supports Python with GTK widget libraries, enabling programs to be written and run on the target device.[161]

Cross-compilers to other languages

There are several compilers/transpilers to high-level object languages, with either unrestricted Python, a restricted subset of Python, or a language similar to Python as the source language:

- Brython,[162] Transcrypt[163][164] and Pyjs (latest release in 2012) compile Python to JavaScript.

- Cython compiles (a superset of) Python to C. The resulting code is also usable with Python via direct C-level API calls into the Python interpreter.
- PyJL compiles/transpiles a subset of Python to "human-readable, maintainable, and high-performance Julia source code".[89] Despite claiming high performance, no tool can claim to do that for *arbitrary* Python code; i.e. it's known not possible to compile to a faster language or machine code. Unless semantics of Python are changed, but in many cases speedup is possible with few or no changes in the Python code. The faster Julia source code can then be used from Python, or compiled to machine code, and based that way.
- Nuitka compiles Python into C.[165] It works with Python 3.4 to 3.12 (and 2.6 and 2.7), for Python's main supported platforms (and Windows 7 or even Windows XP) and for Android. It claims complete support for Python 3.10, some support for 3.11 and 3.12 and experimental support for Python 3.13. It supports macOS including Apple Silicon-based. It's a free compiler, though it also has commercial add-ons (e.g. for hiding source code).
- Numba is used from Python, as a tool (enabled by adding a decorator to relevant Python code), a JIT compiler that translates a subset of Python and NumPy code into fast machine code.
- Pythran compiles a subset of Python 3 to C++ (C++11).[166]
- RPython can be compiled to C, and is used to build the PyPy interpreter of Python.
- The Python → 11l → C++ transpiler[167] compiles a subset of Python 3 to C++ (C++17).

Specialized:

- MyHDL is a Python-based hardware description language (HDL), that converts MyHDL code to Verilog or VHDL code.

Older projects (or not to be used with Python 3.x and latest syntax):

- Google's Grumpy (latest release in 2017) transpiles Python 2 to Go.[168][169][170]
- IronPython allows running Python 2.7 programs (and an alpha, released in 2021, is also available for "Python 3.4, although features and behaviors from later versions may be included"[171]) on the .NET Common Language Runtime.[172]
- Jython compiles Python 2.7 to Java bytecode, allowing the use of the Java libraries from a Python program.[173]
- Pyrex (latest release in 2010) and Shed Skin (latest release in 2013) compile to C and C++ respectively.

Performance

Performance comparison of various Python implementations on a non-numerical (combinatorial) workload was presented at EuroSciPy '13.[174] Python's performance compared to other programming languages is also benchmarked by The Computer Language Benchmarks Game.[175]

There are some ways of optimizing performance due to the slower nature of Python being an interpreted language. These optimizations use different strategies or tools:

- Just-in-time compilation: Compiling code just before it is executed dynamically. These implementations are seen in libraries like Numba and PyPy.
- Static compilation: Code gets compiled into machine code before execution. An example of this is Cython which compiles Python into C.
- Concurrency and Parallelism: Multiple tasks can be run at the same time. Python contains modules like `multiprocessing` to allow for this form of parallelism. Additionally, it helps to overcome limitations of the Global Interpreter Lock (GIL) in CPU tasks.
- Efficient Data Structures: Using data types such as `Set` for membership tests or deque from `collections` for queue operations can also improve performance.

Development

Python's development is conducted largely through the *Python Enhancement Proposal* (PEP) process, the primary mechanism for proposing major new features, collecting community input on issues, and documenting Python design decisions.[176] Python coding style is covered in PEP 8.[177] Outstanding PEPs are reviewed and commented on by the Python community and the steering council.[176]

Enhancement of the language corresponds with the development of the CPython reference implementation. The mailing list python-dev is the primary forum for the language's development. Specific issues were originally discussed in the Roundup bug tracker hosted at by the foundation.[178] In 2022, all issues and discussions were migrated to GitHub.[179] Development originally took place on a self-hosted source-code repository running Mercurial, until Python moved to GitHub in January 2017.[180]

CPython's public releases come in three types, distinguished by which part of the version number is incremented:

- Backward-incompatible versions, where code is expected to break and needs to be manually ported. The first part of the version number is incremented. These releases happen infrequently—version 3.0 was released 8 years after 2.0. According to Guido van Rossum, a version 4.0 is very unlikely to ever happen.[181]
- Major or "feature" releases are largely compatible with the previous version but introduce new features. The second part of the version number is incremented.

Starting with Python 3.9, these releases are expected to happen annually.[182] [183] Each major version is supported by bug fixes for several years after its release.[184]

- Bug fix releases,[185] which introduce no new features, occur about every 3 months and are made when a sufficient number of bugs have been fixed upstream since the last release. Security vulnerabilities are also patched in these releases. The third and final part of the version number is incremented.[185]

Many alpha, beta, and release-candidates are also released as previews and for testing before final releases. Although there is a rough schedule for each release, they are often delayed if the code is not ready. Python's development team monitors the state of the code by running the large unit test suite during development.[186]

The major academic conference on Python is PyCon. There are also special Python mentoring programs, such as PyLadies.

Python 3.12 removed `wstr` meaning Python extensions[187] need to be modified,[188] and 3.10 added pattern matching to the language.[189]

Python 3.12 dropped some outdated modules, and more will be dropped in the future, deprecated as of 3.13; already deprecated array 'u' format code will emit `DeprecationWarning` since 3.13 and will be removed in Python 3.16. The 'w' format code should be used instead. Part of ctypes is also deprecated and `http.server.CGIHTTPRequestHandler` will emit a DeprecationWarning, and will be removed in 3.15. Using that code already has a high potential for both security and functionality bugs. Parts of the typing module are deprecated, e.g. creating a `typing.NamedTuple` class using keyword arguments to denote the fields and such (and more) will be disallowed in Python 3.15.

API documentation generators

Tools that can generate documentation for Python API include pydoc (available as part of the standard library), Sphinx, Pdoc and its forks, Doxygen and Graphviz, among others.[190]

Naming

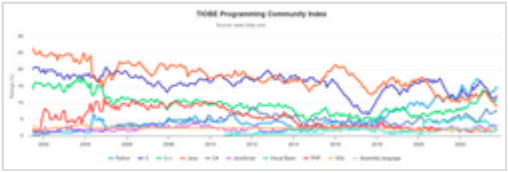
Python's name is derived from the British comedy group Monty Python, whom Python creator Guido van Rossum enjoyed while developing the language. Monty Python references appear frequently in Python code and culture;[191] for example, the metasyntactic variables often used in Python literature are *spam* and *eggs* instead of the traditional *foo* and *bar*.[191][192] The official Python documentation also contains various references to Monty Python routines.[193][194] Users of Python are sometimes referred to as "Pythonistas".[195]

The prefix *Py-* is used to show that something is related to Python. Examples of the use of this prefix in names of Python applications or libraries include Pygame, a binding of Simple DirectMedia Layer to Python (commonly used to create games); PyQt and PyGTK, which bind Qt and GTK to Python respectively; and PyPy, a Python implementation originally written in Python.

Popularity

Since 2003, Python has consistently ranked in the top ten most popular programming languages in the TIOBE Programming Community Index where as of December 2022 it was the most popular language (ahead of C, C++, and Java).[40] It was selected as Programming Language of the Year (for "the highest rise in ratings in a year") in 2007, 2010, 2018, and 2020 (the only language to have done so four times as of 2020[196]). In the TIOBE Index, monthly rankings are based on the volume of searches for programming languages on Google, Amazon, Wikipedia, Bing, and 20 other platforms. According to the accompanying graph, Python has shown a marked upward trend since the early 2000s, eventually passing long-established languages such as C, C++, and Java. This progression can be attributed to Python's readable syntax, comprehensive standard library, and application in data science and machine learning fields[197].

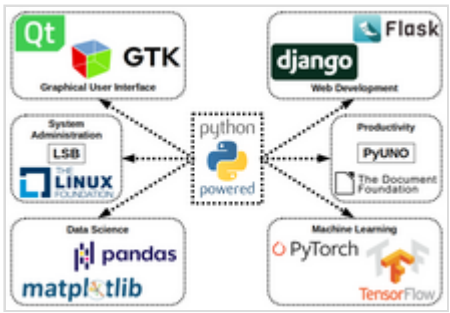
Large organizations that use Python include Wikipedia, Google,[198] Yahoo!,[199] CERN,[200] NASA,[201] Facebook,[202] Amazon, Instagram,[203] Spotify,[204] and some smaller entities like Industrial Light & Magic[205] and ITA.[206] The social news networking site Reddit was written mostly in Python.[207] Organizations that partially use Python include Discord[208] and Baidu.[209]



TIOBE Index Chart showing Python's popularity compared to other programming languages.

Uses

Python can serve as a scripting language for web applications, e.g. via mod_wsgi for the Apache webserver.[210] With Web Server Gateway Interface, a standard API has evolved to facilitate these applications. Web frameworks like Django, Pylons, Pyramid, TurboGears, web2py, Tornado, Flask, Bottle, and Zope support developers in the design and maintenance of complex applications. Pyjs and IronPython can be used to develop the client-side of Ajax-based applications. SQLAlchemy can be used as a data mapper to a relational database. Twisted is a framework to program communications between computers, and is used (for example) by Dropbox.



Python Powered

Libraries such as NumPy, SciPy and Matplotlib allow the effective use of Python in scientific computing,[211][212] with specialized libraries such as Biopython and Astropy providing domain-specific functionality. SageMath is a computer algebra system with a notebook interface programmable in Python: its library covers many aspects of mathematics, including algebra, combinatorics, numerical mathematics, number theory, and calculus.[213] OpenCV has Python bindings with a rich set of features for computer vision and image processing.[214]
Python is commonly used in artificial intelligence projects and machine learning projects with the help of libraries like TensorFlow, Keras, Pytorch, scikit-learn and the Logic language ProbLog.[215][216][217][218][219] As a scripting language with a modular architecture, simple syntax, and rich text processing tools, Python is often used for natural language processing.[220]
The combination of Python and Prolog has proved to be particularly useful for AI applications, with Prolog providing knowledge representation and reasoning capabilities. The Janus system, in particular, exploits the similarities between these two languages, in part because of their use of dynamic typing, and the simple recursive nature of their data structures. Typical applications of this combination include natural language processing, visual query answering, geospatial reasoning, and handling of semantic web data.[221][222] The Natlog system, implemented in Python, uses Definite Clause Grammars (DCGs) as prompt generators for text-to-text generators like GPT3 and text-to-image generators like DALL-E or Stable Diffusion.[223]
Python can also be used for graphical user interface (GUI) by using libraries like Tkinter.[224][225]

Python is embedded in many software products as a scripting language, including in finite element method software such as Abaqus, 3D parametric modelers like FreeCAD, 3D animation packages such as 3ds Max, Blender, Cinema 4D, Lightwave, Houdini, Maya, modo, MotionBuilder, Softimage, the visual effects compositor Nuke, 2D imaging programs like GIMP,[226] Inkscape, Scribus and Paint Shop Pro,[227] and musical notation programs like scorewriter and capella. GNU Debugger uses Python as a pretty printer to show complex structures such as C++ containers. Esri promotes Python as the best choice for writing scripts in ArcGIS.[228] It has also been used in several video games,[229][230] and has been adopted as first of the three available programming languages in Google App Engine, the other two being Java and Go.[231]
Many operating systems include Python as a standard component. It ships with most Linux distributions,[232] AmigaOS 4 (using Python 2.7), FreeBSD (as a package), NetBSD, and OpenBSD (as a package) and can be used from the command line (terminal). Many Linux distributions use installers written in Python: Ubuntu uses the Ubiquity installer, while Red Hat Linux and Fedora Linux use the Anaconda installer. Gentoo Linux uses Python in its package management system, Portage.
Python is used extensively in the information security industry, including in exploit development.[233][234]

Most of the Sugar software for the One Laptop per Child XO, developed at Sugar Labs as of 2008, is written in Python.[235] The Raspberry Pi single-board computer project has adopted Python as its main user-programming language.
LibreOffice includes Python and intends to replace Java with Python. Its Python Scripting Provider is a core feature[236] since Version 4.0 from 7 February 2013.

Languages influenced by Python

Python's design and philosophy have influenced many other programming languages:

- Boo uses indentation, a similar syntax, and a similar object model.[237]
- Cobra uses indentation and a similar syntax, and its *Acknowledgements* document lists Python first among languages that influenced it.[238]
- CoffeeScript, a programming language that cross-compiles to JavaScript, has Python-inspired syntax.
- ECMAScript–JavaScript borrowed iterators and generators from Python.[239]
- GDScript, a scripting language very similar to Python, built-in to the Godot game engine.[240]
- Go is designed for the "speed of working in a dynamic language like Python"[241] and shares the same syntax for slicing arrays.
- Groovy was motivated by the desire to bring the Python design philosophy to Java.[242]
- Julia was designed to be "as usable for general programming as Python".[27]
- Mojo is a non-strict[28][243] superset of Python (e.g. still missing classes, and adding e.g. struct).[244]
- Nim uses indentation and similar syntax.[245]
- Ruby's creator, Yukihiro Matsumoto, has said: "I wanted a scripting language that was more powerful than Perl, and more object-oriented than Python. That's why I decided to design my own language."[246]
- Swift, a programming language developed by Apple, has some Python-inspired syntax.[247]
- Kotlin blends Python and Java features, minimizing boilerplate code for enhanced developer efficiency.[248]

Python's development practices have also been emulated by other languages. For example, the practice of requiring a document describing the rationale for, and issues surrounding, a change to the language (in Python, a PEP) is also used in Tcl,[249] Erlang,[250] and Swift.[251]

See also

Computer programming portal

Free and open-source software portal

- Python syntax and semantics
- pip (package manager)

- List of programming languages
- History of programming languages
- Comparison of programming languages

References

0. "General Python FAQ – Python 3 documentation" (https://docs.python.org/3/faq/general.html#what-is-python). *docs.python.org*. Retrieved 7 July 2024.
0. "Python 0.9.1 part 01/21" (https://www.tuhs.org/Usenet/alt.sources/1991-February/001749.html). alt.sources archives. Archived (https://web.archive.org/web/20210811171015/https://www.tuhs.org/Usenet/alt.sources/1991-February/001749.html) from the original on 11 August 2021. Retrieved 11 August 2021.
0. "Why is Python a dynamic language and also a strongly typed language" (https://wiki.python.org/moin/Why%20is%20Python%20a%20dynamic%20language%20and%20also%20a%20strongly%20typed%20language). *Python Wiki*. Archived (https://web.archive.org/web/20210314173706/https://wiki.python.org/moin/Why%20is%20Python%20a%20dynamic%20language%20and%20also%20a%20strongly%20typed%20language) from the original on 14 March 2021. Retrieved 27 January 2021.
0. "PEP 483 – The Theory of Type Hints" (https://www.python.org/dev/peps/pep-0483/). *Python.org*. Archived (https://web.archive.org/web/20200614153558/https://www.python.org/dev/peps/pep-0483/) from the original on 14 June 2020. Retrieved 14 June 2018.
0. "PEP 11 – CPython platform support | peps.python.org" (https://peps.python.org/pep-0011/). *Python Enhancement Proposals (PEPs)*. Retrieved 22 April 2024.
0. "PEP 738 – Adding Android as a supported platform | peps.python.org" (https://peps.python.org/pep-0738/). *Python Enhancement Proposals (PEPs)*. Retrieved 19 May 2024.
0. "Download Python for Other Platforms" (https://www.python.org/download/other/). *Python.org*. Archived (https://web.archive.org/web/20201127015815/https://www.python.org/download/other/) from the original on 27 November 2020. Retrieved 18 August 2023.
0. "test – Regression tests package for Python – Python 3.7.13 documentation" (https://docs.python.org/3.7/library/test.html?highlight=android#test.support.is_android). *docs.python.org*. Archived (https://web.archive.org/web/20220517151240/https://docs.python.org/3.7/library/test.html?highlight=android#test.support.is_android) from the original on 17 May 2022. Retrieved 17 May 2022.

0. "platform – Access to underlying platform's identifying data – Python 3.10.4 documentation" (https://docs.python.org/3/library/platform.html?highlight=android). *docs.python.org*. Archived (https://web.archive.org/web/20220517150826/https://docs.python.org/3/library/platform.html?highlight=android) from the original on 17 May 2022. Retrieved 17 May 2022.
0. Holth, Moore (30 March 2014). "PEP 0441 – Improving Python ZIP Application Support" (https://www.python.org/dev/peps/pep-0441/). Archived (https://web.archive.org/web/20181226141117/https://www.python.org/dev/peps/pep-0441/%20) from the original on 26 December 2018. Retrieved 12 November 2015.
0. "Starlark Language" (https://docs.bazel.build/versions/master/skylark/language.html). Archived (https://web.archive.org/web/20200615140534/https://docs.bazel.build/versions/master/skylark/language.html) from the original on 15 June 2020. Retrieved 25 May 2019.
0. "Why was Python created in the first place?" (https://docs.python.org/faq/general.html#why-was-python-created-in-the-first-place). *General Python FAQ*. Python Software Foundation. Archived (https://web.archive.org/web/20121024164224/http://docs.python.org/faq/general.html#why-was-python-created-in-the-first-place) from the original on 24 October 2012. Retrieved 22 March 2007. "I had extensive experience with implementing an interpreted language in the ABC group at CWI, and from working with this group I had learned a lot about language design. This is the origin of many Python features, including the use of indentation for statement grouping and the inclusion of very high-level data types (although the details are all different in Python)."
0. "Ada 83 Reference Manual (raise statement)" (http://archive.adaic.com/standards/83lrm/html/lrm-11-03.html#11.3). Archived (https://web.archive.org/web/20191022155758/http://archive.adaic.com/standards/83lrm/html/lrm-11-03.html#11.3) from the original on 22 October 2019. Retrieved 7 January 2020.
0. Kuchling, Andrew M. (22 December 2006). "Interview with Guido van Rossum (July 1998)" (https://web.archive.org/web/20070501105422/http://www.amk.ca/python/writing/gvr-interview). *amk.ca*. Archived from the original (http://www.amk.ca/python/writing/gvr-interview) on 1 May 2007. Retrieved 12 March 2012. "I'd spent a summer at DEC's Systems Research Center, which introduced me to Modula-2+; the Modula-3 final report was being written there at about the same time. What I learned there later showed up in Python's exception handling, modules, and the fact that methods explicitly contain 'self' in their parameter list. String slicing came from Algol-68 and Icon."

0. "itertools – Functions creating iterators for efficient looping – Python 3.7.1 documentation" (https://docs.python.org/3/library/itertools.html). *docs.python.org*. Archived (https://web.archive.org/web/20200614153629/https://docs.python.org/3/library/itertools.html) from the original on 14 June 2020. Retrieved 22 November 2016. "This module implements a number of iterator building blocks inspired by constructs from APL, Haskell, and SML."
0. van Rossum, Guido (1993). "An Introduction to Python for UNIX/C Programmers". *Proceedings of the NLUUG Najaarsconferentie (Dutch UNIX Users Group)*. CiteSeerX 10.1.1.38.2023 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.38.2023). "even though the design of C is far from ideal, its influence on Python is considerable."
0. "Classes" (https://docs.python.org/tutorial/classes.html). *The Python Tutorial*. Python Software Foundation. Archived (https://web.archive.org/web/20121023030209/http://docs.python.org/tutorial/classes.html) from the original on 23 October 2012. Retrieved 20 February 2012. "It is a mixture of the class mechanisms found in C++ and Modula-3"
0. Lundh, Fredrik. "Call By Object" (http://effbot.org/zone/call-by-object.htm). *effbot.org*. Archived (https://web.archive.org/web/20191123043655/http://effbot.org/zone/call-by-object.htm) from the original on 23 November 2019. Retrieved 21 November 2017. "replace "CLU" with "Python", "record" with "instance", and "procedure" with "function or method", and you get a pretty accurate description of Python's object model."
0. Simionato, Michele. "The Python 2.3 Method Resolution Order" (https://www.python.org/download/releases/2.3/mro/). Python Software Foundation. Archived (https://web.archive.org/web/20200820231854/https://www.python.org/download/releases/2.3/mro/) from the original on 20 August 2020. Retrieved 29 July 2014. "The C3 method itself has nothing to do with Python, since it was invented by people working on Dylan and it is described in a paper intended for lispers"
0. Kuchling, A. M. "Functional Programming HOWTO" (https://docs.python.org/howto/functional.html). *Python v2.7.2 documentation*. Python Software Foundation. Archived (https://web.archive.org/web/20121024163217/http://docs.python.org/howto/functional.html) from the original on 24 October 2012. Retrieved 9 February 2012. "List comprehensions and generator expressions [...] are a concise notation for such operations, borrowed from the functional programming language Haskell."
0. Schemenauer, Neil; Peters, Tim; Hetland, Magnus Lie (18 May 2001). "PEP 255 – Simple Generators" (https://www.python.org/dev/peps/pep-0255/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200605012926/https://www.python.org/dev/peps/pep-0255/) from the original on 5 June 2020. Retrieved 9 February 2012.

0. "More Control Flow Tools" (https://docs.python.org/3.2/tutorial/controlflow.html). *Python 3 documentation*. Python Software Foundation. Archived (https://web.archive.org/web/20160604080843/https://docs.python.org/3.2/tutorial/controlflow.html) from the original on 4 June 2016. Retrieved 24 July 2015. "By popular demand, a few features commonly found in functional programming languages like Lisp have been added to Python. With the lambda keyword, small anonymous functions can be created."
0. "re – Regular expression operations – Python 3.10.6 documentation" (https://docs.python.org/3/library/re.html). *docs.python.org*. Archived (https://web.archive.org/web/20180718132241/https://docs.python.org/3/library/re.html) from the original on 18 July 2018. Retrieved 6 September 2022. "This module provides regular expression matching operations similar to those found in Perl."
0. "CoffeeScript" (https://coffeescript.org/). *coffeescript.org*. Archived (https://web.archive.org/web/20200612100004/http://coffeescript.org/) from the original on 12 June 2020. Retrieved 3 July 2018.
0. "Perl and Python influences in JavaScript" (http://www.2ality.com/2013/02/javascript-influences.html). *www.2ality.com*. 24 February 2013. Archived (https://web.archive.org/web/20181226141121/http://2ality.com/2013/02/javascript-influences.html%0A) from the original on 26 December 2018. Retrieved 15 May 2015.
0. Rauschmayer, Axel. "Chapter 3: The Nature of JavaScript; Influences" (http://speakingjs.com/es5/ch03.html). *O'Reilly, Speaking JavaScript*. Archived (https://web.archive.org/web/20181226141123/http://speakingjs.com/es5/ch03.html%0A) from the original on 26 December 2018. Retrieved 15 May 2015.
0. "Why We Created Julia" (https://julialang.org/blog/2012/02/why-we-created-julia). *Julia website*. February 2012. Archived (https://web.archive.org/web/20200502144010/https://julialang.org/blog/2012/02/why-we-created-julia/) from the original on 2 May 2020. Retrieved 5 June 2014. "We want something as usable for general programming as Python [...]"
0. Krill, Paul (4 May 2023). "Mojo language marries Python and MLIR for AI development" (https://www.infoworld.com/article/3695588/mojo-language-marries-python-and-mlir-for-ai-development.html). *InfoWorld*. Archived (https://web.archive.org/web/20230505064554/https://www.infoworld.com/article/3695588/mojo-language-marries-python-and-mlir-for-ai-development.html) from the original on 5 May 2023. Retrieved 5 May 2023.
0. Ring Team (4 December 2017). "Ring and other languages" (http://ring-lang.sourceforge.net/doc1.6/introduction.html#ring-and-other-languages). *ring-lang.net*. ring-lang. Archived (https://web.archive.org/web/20181225175312/http://ring-lang.sourceforge.net/doc1.6/introduction.html#ring-and-other-languages) from the original on 25 December 2018. Retrieved 4 December 2017.

0. Bini, Ola (2007). *Practical JRuby on Rails Web 2.0 Projects: bringing Ruby on Rails to the Java platform* (https://archive.org/details/practicaljrubyon0000bini/page/3). Berkeley: APress. p. 3 (https://archive.org/details/practicaljrubyon0000bini/page/3). ISBN 978-1-59059-881-8.
0. Lattner, Chris (3 June 2014). "Chris Lattner's Homepage" (http://nondot.org/sabre/). Chris Lattner. Archived (https://web.archive.org/web/20181225175312/http://nondot.org/sabre/) from the original on 25 December 2018. Retrieved 3 June 2014. "The Swift language is the product of tireless effort from a team of language experts, documentation gurus, compiler optimization ninjas, and an incredibly important internal dogfooding group who provided feedback to help refine and battle-test ideas. Of course, it also greatly benefited from the experiences hard-won by many other languages in the field, drawing ideas from Objective-C, Rust, Haskell, Ruby, Python, C#, CLU, and far too many others to list."
0. "V documentation (Introduction)" (https://github.com/vlang/v/blob/master/doc/docs.md#introduction). *GitHub*. Retrieved 24 December 2024.
0. Kuhlman, Dave. "A Python Book: Beginning Python, Advanced Python, and Python Exercises" (https://web.archive.org/web/20120623165941/http://cutter.rexx.com/~dkuhlman/python_book_01.html). Section 1.1. Archived from the original (https://www.davekuhlman.org/python_book_01.pdf) (PDF) on 23 June 2012.
0. "About Python" (https://www.python.org/about). Python Software Foundation. Archived (https://web.archive.org/web/20120420010049/http://www.python.org/about/) from the original on 20 April 2012. Retrieved 24 April 2012., second section "Fans of Python use the phrase "batteries included" to describe the standard library, which covers everything from asynchronous processing to zip files."
0. "PEP 206 – Python Advanced Library" (https://www.python.org/dev/peps/pep-0206/). *Python.org*. Archived (https://web.archive.org/web/20210505003659/https://www.python.org/dev/peps/pep-0206/) from the original on 5 May 2021. Retrieved 11 October 2021.
0. Rossum, Guido Van (20 January 2009). "The History of Python: A Brief Timeline of Python" (https://python-history.blogspot.com/2009/01/brief-timeline-of-python.html). *The History of Python*. Archived (https://web.archive.org/web/20200605032200/https://python-history.blogspot.com/2009/01/brief-timeline-of-python.html) from the original on 5 June 2020. Retrieved 5 March 2021.
0. Peterson, Benjamin (20 April 2020). "Python 2.7.18, the last release of Python 2" (https://pythoninsider.blogspot.com/2020/04/python-2718-last-release-of-python-2.html). *Python Insider*. Archived (https://web.archive.org/web/20200426204118/https://pythoninsider.blogspot.com/2020/04/python-2718-last-release-of-python-2.html) from the original on 26 April 2020. Retrieved 27 April 2020.

0. "Stack Overflow Developer Survey 2022" (https://survey.stackoverflow.co/2022/). *Stack Overflow*. Archived (https://web.archive.org/web/20220627175307/https://survey.stackoverflow.co/2022/) from the original on 27 June 2022. Retrieved 12 August 2022.
0. "The State of Developer Ecosystem in 2020 Infographic" (https://www.jetbrains.com/lp/devecosystem-2020/). *JetBrains: Developer Tools for Professionals and Teams*. Archived (https://web.archive.org/web/20210301062411/https://www.jetbrains.com/lp/devecosystem-2020/) from the original on 1 March 2021. Retrieved 5 March 2021.
0. "TIOBE Index" (https://www.tiobe.com/tiobe-index/). TIOBE. Archived (https://web.archive.org/web/20180225101948/https://www.tiobe.com/tiobe-index/) from the original on 25 February 2018. Retrieved 3 January 2023. "The TIOBE Programming Community index is an indicator of the popularity of programming languages" Updated as required.
0. "PYPL PopularitY of Programming Language index" (https://pypl.github.io/PYPL.html). *pypl.github.io*. Archived (https://web.archive.org/web/20170314232030/https://pypl.github.io/PYPL.html) from the original on 14 March 2017. Retrieved 26 March 2021.
0. Venners, Bill (13 January 2003). "The Making of Python" (http://www.artima.com/intv/pythonP.html). *Artima Developer*. Artima. Archived (https://web.archive.org/web/20160901183332/http://www.artima.com/intv/pythonP.html) from the original on 1 September 2016. Retrieved 22 March 2007.
0. van Rossum, Guido (29 August 2000). "SETL (was: Lukewarm about range literals)" (https://mail.python.org/pipermail/python-dev/2000-August/008881.html). *Python-Dev* (Mailing list). Archived (https://web.archive.org/web/20180714064019/https://mail.python.org/pipermail/python-dev/2000-August/008881.html) from the original on 14 July 2018. Retrieved 13 March 2011.
0. van Rossum, Guido (20 January 2009). "A Brief Timeline of Python" (https://python-history.blogspot.com/2009/01/brief-timeline-of-python.html). *The History of Python*. Archived (https://web.archive.org/web/20200605032200/https://python-history.blogspot.com/2009/01/brief-timeline-of-python.html) from the original on 5 June 2020. Retrieved 20 January 2009.
0. Fairchild, Carlie (12 July 2018). "Guido van Rossum Stepping Down from Role as Python's Benevolent Dictator For Life" (https://www.linuxjournal.com/content/guido-van-rossum-stepping-down-role-pythons-benevolent-dictator-life). *Linux Journal*. Archived (https://web.archive.org/web/20180713192427/https://www.linuxjournal.com/content/guido-van-rossum-stepping-down-role-pythons-benevolent-dictator-life) from the original on 13 July 2018. Retrieved 13 July 2018.

0. "PEP 8100" (https://www.python.org/dev/peps/pep-8100/). Python Software Foundation. Archived (https://web.archive.org/web/20200604235027/https://www.python.org/dev/peps/pep-8100/) from the original on 4 June 2020. Retrieved 4 May 2019.
0. "PEP 13 – Python Language Governance" (https://www.python.org/dev/peps/pep-0013/). *Python.org*. Archived (https://web.archive.org/web/20210527000035/https://www.python.org/dev/peps/pep-0013/) from the original on 27 May 2021. Retrieved 25 August 2021.
0. Briggs, Jason R.; Lipovača, Miran (2013). *Python for kids: a playful introduction to programming*. San Francisco, Calif: No Starch Press. ISBN 978-1-59327-407-8.
0. Kuchling, A. M.; Zadka, Moshe (16 October 2000). "What's New in Python 2.0" (https://docs.python.org/whatsnew/2.0.html). Python Software Foundation. Archived (https://web.archive.org/web/20121023112045/http://docs.python.org/whatsnew/2.0.html) from the original on 23 October 2012. Retrieved 11 February 2012.
0. "PEP 373 – Python 2.7 Release Schedule" (https://legacy.python.org/dev/peps/pep-0373/). *python.org*. Archived (https://web.archive.org/web/20200519075520/https://legacy.python.org/dev/peps/pep-0373/) from the original on 19 May 2020. Retrieved 9 January 2017.
0. "PEP 466 – Network Security Enhancements for Python 2.7.x" (https://www.python.org/dev/peps/pep-0466/). *python.org*. Archived (https://web.archive.org/web/20200604232833/https://www.python.org/dev/peps/pep-0466/) from the original on 4 June 2020. Retrieved 9 January 2017.
0. "Sunsetting Python 2" (https://www.python.org/doc/sunset-python-2/). *Python.org*. Archived (https://web.archive.org/web/20200112080903/https://www.python.org/doc/sunset-python-2/) from the original on 12 January 2020. Retrieved 22 September 2019.
0. "PEP 373 – Python 2.7 Release Schedule" (https://www.python.org/dev/peps/pep-0373/). *Python.org*. Archived (https://web.archive.org/web/20200113033257/https://www.python.org/dev/peps/pep-0373/) from the original on 13 January 2020. Retrieved 22 September 2019.
0. mattip (25 December 2023). "PyPy v7.3.14 release" (https://www.pypy.org/posts/2023/12/pypy-v7314-release.html). *PyPy*. Archived (https://web.archive.org/web/20240105132820/https://www.pypy.org/posts/2023/12/pypy-v7314-release.html) from the original on 5 January 2024. Retrieved 5 January 2024.
0. Langa, Łukasz (17 May 2022). "Python 3.9.13 is now available" (https://pythoninsider.blogspot.com/2022/05/python-3913-is-now-available.html). *Python Insider*. Archived (https://web.archive.org/web/20220517173546/https://pythoninsider.blogspot.com/2022/05/python-3913-is-now-available.html) from the original on 17 May 2022. Retrieved 21 May 2022.

0. "Status of Python versions" (https://devguide.python.org/versions/). *Python Developer's Guide*. Retrieved 7 October 2024.
0. "Python" (https://endoflife.date/python). *endoflife.date*. 8 October 2024. Retrieved 20 November 2024.
0. "CVE-2021-3177" (https://access.redhat.com/security/cve/cve-2021-3177). *Red Hat Customer Portal*. Archived (https://web.archive.org/web/20210306183700/https://access.redhat.com/security/cve/cve-2021-3177) from the original on 6 March 2021. Retrieved 26 February 2021.
0. "CVE-2021-3177" (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-3177). *CVE*. Archived (https://web.archive.org/web/20210227192918/https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-3177) from the original on 27 February 2021. Retrieved 26 February 2021.
0. "CVE-2021-23336" (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-23336). *CVE*. Archived (https://web.archive.org/web/20210224160700/https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-23336) from the original on 24 February 2021. Retrieved 26 February 2021.
0. "Built-in Types" (https://docs.python.org/3/library/stdtypes.html#types-union).
0. corbet (24 October 2022). "Python 3.11 released [LWN.net]" (https://lwn.net/Articles/912216/). *lwn.net*. Retrieved 15 November 2022.
0. "What's New In Python 3.13" (https://docs.python.org/3.13/whatsnew/3.13.html#experimental-jit-compiler). *Python documentation*. Retrieved 30 April 2024.
0. "PEP 667 – Consistent views of namespaces | peps.python.org" (https://peps.python.org/pep-0667/). *Python Enhancement Proposals (PEPs)*. Retrieved 7 October 2024.
0. "PEP 703 – Making the GIL Optional in CPython" (https://peps.python.org/pep-0703/). *Python Enhancement Proposals (PEPs)*. Retrieved 30 March 2025.
0. Wouters, Thomas (9 April 2024). "Python Insider: Python 3.12.3 and 3.13.0a6 released" (https://pythoninsider.blogspot.com/2024/04/python-3123-and-3130a6-released.html). *Python Insider*. Retrieved 29 April 2024.
0. "PEP 594 – Removing dead batteries from the standard library" (https://peps.python.org/pep-0594/). *Python Enhancement Proposals*. Python Softtware Foundation. 20 May 2019.
0. "PEP 761 – Deprecating PGP signatures for CPython artifacts | peps.python.org" (https://peps.python.org/pep-0761/). *Python Enhancement Proposals (PEPs)*. Retrieved 6 January 2025.
0. "PEP 749 – Implementing PEP 649 | peps.python.org" (https://peps.python.org/pep-0749/). *Python Enhancement Proposals (PEPs)*. Retrieved 20 November 2024.

0. "PEP 711: PyBI: a standard format for distributing Python Binaries" (https://discuss.python.org/t/pep-711-pybi-a-standard-format-for-distributing-python-binaries/25547). *Discussions on Python.org*. 7 April 2023. Retrieved 20 November 2024.
0. "PEP 686 – Make UTF-8 mode default | peps.python.org" (https://peps.python.org/pep-0686/). *Python Enhancement Proposals (PEPs)*. Retrieved 20 November 2024.
0. The Cain Gang Ltd. "Python Metaclasses: Who? Why? When?" (https://web.archive.org/web/20090530030205/http://www.python.org/community/pycon/dc2004/papers/24/metaclasses-pycon.pdf) (PDF). Archived from the original (https://www.python.org/community/pycon/dc2004/papers/24/metaclasses-pycon.pdf) (PDF) on 30 May 2009. Retrieved 27 June 2009.
0. "3.3. Special method names" (https://docs.python.org/3.0/reference/datamodel.html#special-method-names). *The Python Language Reference*. Python Software Foundation. Archived (https://web.archive.org/web/20181215123146/https://docs.python.org/3.0/reference/datamodel.html#special-method-names) from the original on 15 December 2018. Retrieved 27 June 2009.
0. "PyDBC: method preconditions, method postconditions and class invariants for Python" (http://www.nongnu.org/pydbc/). Archived (https://web.archive.org/web/20191123231931/http://www.nongnu.org/pydbc/) from the original on 23 November 2019. Retrieved 24 September 2011.
0. "Contracts for Python" (http://www.wayforward.net/pycontract/). Archived (https://web.archive.org/web/20200615173404/http://www.wayforward.net/pycontract/) from the original on 15 June 2020. Retrieved 24 September 2011.
0. "PyDatalog" (https://sites.google.com/site/pydatalog/). Archived (https://web.archive.org/web/20200613160231/https://sites.google.com/site/pydatalog/) from the original on 13 June 2020. Retrieved 22 July 2012.
0. "Glue It All Together With Python" (https://www.python.org/doc/essays/omg-darpa-mcc-position/). *Python.org*. Retrieved 30 September 2024.
0. "Extending and Embedding the Python Interpreter: Reference Counts" (https://docs.python.org/extending/extending.html#reference-counts). Docs.python.org. Archived (https://web.archive.org/web/20121018063230/http://docs.python.org/extending/extending.html#reference-counts) from the original on 18 October 2012. Retrieved 5 June 2020. "Since Python makes heavy use of `malloc()` and `free()`, it needs a strategy to avoid memory leaks as well as the use of freed memory. The chosen method is called *reference counting*."
0. Hettinger, Raymond (30 January 2002). "PEP 289 – Generator Expressions" (https://www.python.org/dev/peps/pep-0289/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200614153717/https://www.python.org/dev/peps/pep-0289/) from the original on 14 June 2020. Retrieved 19 February 2012.

0. "6.5 itertools – Functions creating iterators for efficient looping" (https://docs.python.org/3/library/itertools.html). Docs.python.org. Archived (https://web.archive.org/web/20200614153629/https://docs.python.org/3/library/itertools.html) from the original on 14 June 2020. Retrieved 22 November 2016.
0. Peters, Tim (19 August 2004). "PEP 20 – The Zen of Python" (https://www.python.org/dev/peps/pep-0020/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20181226141127/https://www.python.org/dev/peps/pep-0020/) from the original on 26 December 2018. Retrieved 24 November 2008.
0. Lutz, Mark (January 2022). "Python Changes 2014+" (https://learning-python.com/python-changes-2014-plus.html). *Learning Python*. Archived (https://web.archive.org/web/20240315075935/https://learning-python.com/python-changes-2014-plus.html) from the original on 15 March 2024. Retrieved 25 February 2024.
0. "Confusion regarding a rule in The Zen of Python" (https://discuss.python.org/t/confusion-regarding-a-rule-in-the-zen-of-python/15927). *Python Help - Discussions on Python.org*. 3 May 2022. Archived (https://web.archive.org/web/20240225221142/https://discuss.python.org/t/confusion-regarding-a-rule-in-the-zen-of-python/15927) from the original on 25 February 2024. Retrieved 25 February 2024.
0. Ambi, Chetan (4 July 2021). "The Most Controversial Python Walrus Operator" (https://pythonsimplified.com/the-most-controversial-python-walrus-operator/). *Python Simplified*. Archived (https://web.archive.org/web/20230827154931/https://pythonsimplified.com/the-most-controversial-python-walrus-operator/) from the original on 27 August 2023. Retrieved 5 February 2024.
0. Grifski, Jeremy (24 May 2020). "The Controversy Behind The Walrus Operator in Python" (https://therenegadecoder.com/code/the-controversy-behind-the-walrus-operator-in-python/). *The Renegade Coder*. Archived (https://web.archive.org/web/20231228135749/https://therenegadecoder.com/code/the-controversy-behind-the-walrus-operator-in-python/) from the original on 28 December 2023. Retrieved 25 February 2024.
0. Bader, Dan. "Python String Formatting Best Practices" (https://realpython.com/python-string-formatting/). *Real Python*. Archived (https://web.archive.org/web/20240218083506/https://realpython.com/python-string-formatting/) from the original on 18 February 2024. Retrieved 25 February 2024.
0. Martelli, Alex; Ravenscroft, Anna; Ascher, David (2005). *Python Cookbook, 2nd Edition* (http://shop.oreilly.com/product/9780596007973.do). O'Reilly Media. p. 230. ISBN 978-0-596-00797-3. Archived (https://web.archive.org/web/20200223171254/http://shop.oreilly.com/product/9780596007973.do) from the original on 23 February 2020. Retrieved 14 November 2015.

0. "Python Culture" (https://web.archive.org/web/20140130021902/http://ebeab.com/2014/01/21/python-culture/). *ebeab*. 21 January 2014. Archived from the original (http://ebeab.com/2014/01/21/python-culture/) on 30 January 2014.
0. "Transpiling Python to Julia using PyJL" (https://web.ist.utl.pt/antonio.menezes.leitao/ADA/documents/publications_docs/2022_TranspilingPythonToJuliaUsingPyJL.pdf) (PDF). Archived (https://web.archive.org/web/20231119071525/https://web.ist.utl.pt/antonio.menezes.leitao/ADA/documents/publications_docs/2022_TranspilingPythonToJuliaUsingPyJL.pdf) (PDF) from the original on 19 November 2023. Retrieved 20 September 2023. "After manually modifying one line of code by specifying the necessary type information, we obtained a speedup of 52.6×, making the translated Julia code 19.5× faster than the original Python code."
0. "Why is it called Python?" (https://docs.python.org/3/faq/general.html#why-is-it-called-python). *General Python FAQ*. Docs.python.org. Archived (https://web.archive.org/web/20121024164224/http://docs.python.org/faq/general.html#why-is-it-called-python) from the original on 24 October 2012. Retrieved 3 January 2023.
0. "15 Ways Python Is a Powerful Force on the Web" (https://web.archive.org/web/20190511065650/http://insidetech.monster.com/training/articles/8114-15-ways-python-is-a-powerful-force-on-the-web). Archived from the original (https://insidetech.monster.com/training/articles/8114-15-ways-python-is-a-powerful-force-on-the-web) on 11 May 2019. Retrieved 3 July 2018.
0. "pprint – Data pretty printer – Python 3.11.0 documentation" (https://docs.python.org/3/library/pprint.html). *docs.python.org*. Archived (https://web.archive.org/web/20210122224848/https://docs.python.org/3/library/pprint.html) from the original on 22 January 2021. Retrieved 5 November 2022. "stuff=['spam', 'eggs', 'lumberjack', 'knights', 'ni']"
0. "Code Style – The Hitchhiker's Guide to Python" (https://docs.python-guide.org/writing/style). *docs.python-guide.org*. Archived (https://web.archive.org/web/20210127154341/https://docs.python-guide.org/writing/style/) from the original on 27 January 2021. Retrieved 20 January 2021.
0. "Is Python a good language for beginning programmers?" (https://docs.python.org/faq/general.html#is-python-a-good-language-for-beginning-programmers). *General Python FAQ*. Python Software Foundation. Archived (https://web.archive.org/web/20121024164224/http://docs.python.org/faq/general.html#is-python-a-good-language-for-beginning-programmers) from the original on 24 October 2012. Retrieved 21 March 2007.
0. "Myths about indentation in Python" (https://web.archive.org/web/20180218162410/http://www.secnetix.de/~olli/Python/block_indentation.hawk). Secnetix.de. Archived from the original (http://www.secnetix.de/~olli/Python/block_indentation.hawk) on 18 February 2018. Retrieved 19 April 2011.

0. Guttag, John V. (12 August 2016). *Introduction to Computation and Programming Using Python: With Application to Understanding Data*. MIT Press. ISBN 978-0-262-52962-4.
0. "PEP 8 – Style Guide for Python Code" (https://www.python.org/dev/peps/pep-0008/). *Python.org*. Archived (https://web.archive.org/web/20190417223549/https://www.python.org/dev/peps/pep-0008/) from the original on 17 April 2019. Retrieved 26 March 2019.
0. "8. Errors and Exceptions – Python 3.12.0a0 documentation" (https://docs.python.org/3.11/tutorial/errors.html). *docs.python.org*. Archived (https://web.archive.org/web/20220509145745/https://docs.python.org/3.11/tutorial/errors.html) from the original on 9 May 2022. Retrieved 9 May 2022.
0. "Highlights: Python 2.5" (https://www.python.org/download/releases/2.5/highlights/). *Python.org*. Archived (https://web.archive.org/web/20190804120408/https://www.python.org/download/releases/2.5/highlights/) from the original on 4 August 2019. Retrieved 20 March 2018.
0. van Rossum, Guido (22 April 2009). "Tail Recursion Elimination" (http://neopythonic.blogspot.be/2009/04/tail-recursion-elimination.html). Neopythonic.blogspot.be. Archived (https://web.archive.org/web/20180519225253/http://neopythonic.blogspot.be/2009/04/tail-recursion-elimination.html) from the original on 19 May 2018. Retrieved 3 December 2012.
0. van Rossum, Guido (9 February 2006). "Language Design Is Not Just Solving Puzzles" (http://www.artima.com/weblogs/viewpost.jsp?thread=147358). *Artima forums*. Artima. Archived (https://web.archive.org/web/20200117182525/https://www.artima.com/weblogs/viewpost.jsp?thread=147358) from the original on 17 January 2020. Retrieved 21 March 2007.
0. van Rossum, Guido; Eby, Phillip J. (10 May 2005). "PEP 342 – Coroutines via Enhanced Generators" (https://www.python.org/dev/peps/pep-0342/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200529003739/https://www.python.org/dev/peps/pep-0342/) from the original on 29 May 2020. Retrieved 19 February 2012.
0. "PEP 380" (https://www.python.org/dev/peps/pep-0380/). Python.org. Archived (https://web.archive.org/web/20200604233821/https://www.python.org/dev/peps/pep-0380/) from the original on 4 June 2020. Retrieved 3 December 2012.
0. "division" (https://docs.python.org). *python.org*. Archived (https://web.archive.org/web/20060720033244/http://docs.python.org/) from the original on 20 July 2006. Retrieved 30 July 2014.
0. "PEP 0465 – A dedicated infix operator for matrix multiplication" (https://www.python.org/dev/peps/pep-0465/). *python.org*. Archived (https://web.archive.org/web/20200604224255/https://www.python.org/dev/peps/pep-0465/) from the original on 4 June 2020. Retrieved 1 January 2016.

0. "Python 3.5.1 Release and Changelog" (https://www.python.org/downloads/release/python-351/). *python.org*. Archived (https://web.archive.org/web/20200514034938/https://www.python.org/downloads/release/python-351/) from the original on 14 May 2020. Retrieved 1 January 2016.
0. "What's New in Python 3.8" (https://docs.python.org/3.8/whatsnew/3.8.html). Archived (https://web.archive.org/web/20200608124345/https://docs.python.org/3.8/whatsnew/3.8.html) from the original on 8 June 2020. Retrieved 14 October 2019.
0. van Rossum, Guido; Hettinger, Raymond (7 February 2003). "PEP 308 – Conditional Expressions" (https://www.python.org/dev/peps/pep-0308/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20160313113147/https://www.python.org/dev/peps/pep-0308/) from the original on 13 March 2016. Retrieved 13 July 2011.
0. "4. Built-in Types – Python 3.6.3rc1 documentation" (https://docs.python.org/3/library/stdtypes.html#tuple). *python.org*. Archived (https://web.archive.org/web/20200614194325/https://docs.python.org/3/library/stdtypes.html#tuple) from the original on 14 June 2020. Retrieved 1 October 2017.
0. "5.3. Tuples and Sequences – Python 3.7.1rc2 documentation" (https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences). *python.org*. Archived (https://web.archive.org/web/20200610050047/https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences) from the original on 10 June 2020. Retrieved 17 October 2018.
0. "PEP 498 – Literal String Interpolation" (https://www.python.org/dev/peps/pep-0498/). *python.org*. Archived (https://web.archive.org/web/20200615184141/https://www.python.org/dev/peps/pep-0498/) from the original on 15 June 2020. Retrieved 8 March 2017.
0. "Why must 'self' be used explicitly in method definitions and calls?" (https://docs.python.org/faq/design.html#why-must-self-be-used-explicitly-in-method-definitions-and-calls). *Design and History FAQ*. Python Software Foundation. Archived (https://web.archive.org/web/20121024164243/http://docs.python.org/faq/design.html#why-must-self-be-used-explicitly-in-method-definitions-and-calls) from the original on 24 October 2012. Retrieved 19 February 2012.
0. Sweigart, Al (2020). *Beyond the Basic Stuff with Python: Best Practices for Writing Clean Code* (https://books.google.com/books?id=7GUKEAAAQBAJ&pg=PA322). No Starch Press. p. 322. ISBN 978-1-59327-966-0. Archived (https://web.archive.org/web/20210813194312/https://books.google.com/books?id=7GUKEAAAQBAJ&pg=PA322) from the original on 13 August 2021. Retrieved 7 July 2021.

0. "The Python Language Reference, section 3.3. New-style and classic classes, for release 2.7.1" (https://web.archive.org/web/20121026063834/http://docs.python.org/reference/datamodel.html#new-style-and-classic-classes). Archived from the original (https://docs.python.org/reference/datamodel.html#new-style-and-classic-classes) on 26 October 2012. Retrieved 12 January 2011.
0. "PEP 484 – Type Hints | peps.python.org" (https://peps.python.org/pep-0484/). *peps.python.org*. Archived (https://web.archive.org/web/20231127205023/https://peps.python.org/pep-0484/) from the original on 27 November 2023. Retrieved 29 November 2023.
0. "typing — Support for type hints" (https://docs.python.org/3/library/typing.html). *Python documentation*. Python Software Foundation. Archived (https://web.archive.org/web/20200221184042/https://docs.python.org/3/library/typing.html) from the original on 21 February 2020. Retrieved 22 December 2023.
0. "mypy – Optional Static Typing for Python" (http://mypy-lang.org/). Archived (https://web.archive.org/web/20200606192012/http://mypy-lang.org/) from the original on 6 June 2020. Retrieved 28 January 2017.
0. "Introduction" (https://mypyc.readthedocs.io/en/latest/introduction.html). *mypyc.readthedocs.io*. Archived (https://web.archive.org/web/20231222000457/https://mypyc.readthedocs.io/en/latest/introduction.html) from the original on 22 December 2023. Retrieved 22 December 2023.
0. "15. Floating Point Arithmetic: Issues and Limitations – Python 3.8.3 documentation" (https://docs.python.org/3.8/tutorial/floatingpoint.html#representation-error). *docs.python.org*. Archived (https://web.archive.org/web/20200606113842/https://docs.python.org/3.8/tutorial/floatingpoint.html#representation-error) from the original on 6 June 2020. Retrieved 6 June 2020. "Almost all machines today (November 2000) use IEEE-754 floating point arithmetic, and almost all platforms map Python floats to IEEE-754 "double precision"."
0. Zadka, Moshe; van Rossum, Guido (11 March 2001). "PEP 237 – Unifying Long Integers and Integers" (https://www.python.org/dev/peps/pep-0237/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200528063237/https://www.python.org/dev/peps/pep-0237/) from the original on 28 May 2020. Retrieved 24 September 2011.
0. "Built-in Types" (https://docs.python.org/3/library/stdtypes.html#typesseq-range). Archived (https://web.archive.org/web/20200614194325/https://docs.python.org/3/library/stdtypes.html#typesseq-range) from the original on 14 June 2020. Retrieved 3 October 2019.

0. "PEP 465 – A dedicated infix operator for matrix multiplication" (https://legacy.python.org/dev/peps/pep-0465/). *python.org*. Archived (https://web.archive.org/web/20200529200310/https://legacy.python.org/dev/peps/pep-0465/) from the original on 29 May 2020. Retrieved 3 July 2018.
0. Zadka, Moshe; van Rossum, Guido (11 March 2001). "PEP 238 – Changing the Division Operator" (https://www.python.org/dev/peps/pep-0238/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200528115550/https://www.python.org/dev/peps/pep-0238/) from the original on 28 May 2020. Retrieved 23 October 2013.
0. "Why Python's Integer Division Floors" (https://python-history.blogspot.com/2010/08/why-pythons-integer-division-floors.html). 24 August 2010. Archived (https://web.archive.org/web/20200605151500/https://python-history.blogspot.com/2010/08/why-pythons-integer-division-floors.html) from the original on 5 June 2020. Retrieved 25 August 2010.
0. "round" (https://docs.python.org/py3k/library/functions.html#round), *The Python standard library, release 3.2, §2: Built-in functions*, archived (https://web.archive.org/web/20121025141808/http://docs.python.org/py3k/library/functions.html#round) from the original on 25 October 2012, retrieved 14 August 2011
0. "round" (https://docs.python.org/library/functions.html#round), *The Python standard library, release 2.7, §2: Built-in functions*, archived (https://web.archive.org/web/20121027081602/http://docs.python.org/library/functions.html#round) from the original on 27 October 2012, retrieved 14 August 2011
0. Beazley, David M. (2009). *Python Essential Reference* (https://archive.org/details/pythonessentialr00beaz_036) (4th ed.). Addison-Wesley Professional. p. 66 (https://archive.org/details/pythonessentialr00beaz_036/page/n90). ISBN 9780672329784.
0. Kernighan, Brian W.; Ritchie, Dennis M. (1988). *The C Programming Language* (2nd ed.). p. 206 (https://archive.org/details/cprogramminglang00bria/page/206).
0. Batista, Facundo (17 October 2003). "PEP 327 – Decimal Data Type" (https://www.python.org/dev/peps/pep-0327/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200604234830/https://www.python.org/dev/peps/pep-0327/) from the original on 4 June 2020. Retrieved 24 November 2008.
0. "What's New in Python 2.6" (https://docs.python.org/2.6/whatsnew/2.6.html). *Python v2.6.9 documentation*. 29 October 2013. Archived (https://web.archive.org/web/20191223213856/https://docs.python.org/2.6/whatsnew/2.6.html) from the original on 23 December 2019. Retrieved 26 September 2015.

0. "10 Reasons Python Rocks for Research (And a Few Reasons it Doesn't) – Hoyt Koepke" (https://web.archive.org/web/20200531211840/https://www.stat.washington.edu/~hoytak/blog/whypython.html). *University of Washington Department of Statistics*. Archived from the original (https://www.stat.washington.edu/~hoytak/blog/whypython.html) on 31 May 2020. Retrieved 3 February 2019.
0. Shell, Scott (17 June 2014). "An introduction to Python for scientific computing" (https://engineering.ucsb.edu/~shell/che210d/python.pdf) (PDF). Archived (https://web.archive.org/web/20190204014642/https://engineering.ucsb.edu/~shell/che210d/python.pdf) (PDF) from the original on 4 February 2019. Retrieved 3 February 2019.
0. Piotrowski, Przemyslaw (July 2006). "Build a Rapid Web Development Environment for Python Server Pages and Oracle" (http://www.oracle.com/technetwork/articles/piotrowski-pythoncore-084049.html). *Oracle Technology Network*. Oracle. Archived (https://web.archive.org/web/20190402124435/https://www.oracle.com/technetwork/articles/piotrowski-pythoncore-084049.html) from the original on 2 April 2019. Retrieved 12 March 2012.
0. Eby, Phillip J. (7 December 2003). "PEP 333 – Python Web Server Gateway Interface v1.0" (https://www.python.org/dev/peps/pep-0333/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200614170344/https://www.python.org/dev/peps/pep-0333/) from the original on 14 June 2020. Retrieved 19 February 2012.
0. "PyPI" (https://pypi.org/). *PyPI*. 13 March 2025. Archived (https://web.archive.org/web/20250222013445/https://pypi.org/) from the original on 22 February 2025.
0. Enthought, Canopy. "Canopy" (https://web.archive.org/web/20170715151703/https://www.enthought.com/products/canopy/). *www.enthought.com*. Archived from the original (https://www.enthought.com/products/canopy/) on 15 July 2017. Retrieved 20 August 2016.
0. "PEP 7 – Style Guide for C Code | peps.python.org" (https://peps.python.org/pep-0007/). *peps.python.org*. Archived (https://web.archive.org/web/20220424202827/https://peps.python.org/pep-0007/) from the original on 24 April 2022. Retrieved 28 April 2022.
0. "4. Building C and C++ Extensions – Python 3.9.2 documentation" (https://docs.python.org/3/extending/building.html). *docs.python.org*. Archived (https://web.archive.org/web/20210303002519/https://docs.python.org/3/extending/building.html) from the original on 3 March 2021. Retrieved 1 March 2021.

0. van Rossum, Guido (5 June 2001). "PEP 7 – Style Guide for C Code" (https://www.python.org/dev/peps/pep-0007/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200601203908/https://www.python.org/dev/peps/pep-0007/) from the original on 1 June 2020. Retrieved 24 November 2008.
0. "CPython byte code" (https://docs.python.org/3/library/dis.html#python-bytecode-instructions). Docs.python.org. Archived (https://web.archive.org/web/20200605151542/https://docs.python.org/3/library/dis.html#python-bytecode-instructions) from the original on 5 June 2020. Retrieved 16 February 2016.
0. "Python 2.5 internals" (http://www.troeger.eu/teaching/pythonvm08.pdf) (PDF). Archived (https://web.archive.org/web/20120806094951/http://www.troeger.eu/teaching/pythonvm08.pdf) (PDF) from the original on 6 August 2012. Retrieved 19 April 2011.
0. "Changelog – Python 3.9.0 documentation" (https://docs.python.org/release/3.9.0/whatsnew/changelog.html#changelog). *docs.python.org*. Archived (https://web.archive.org/web/20210207001142/https://docs.python.org/release/3.9.0/whatsnew/changelog.html#changelog) from the original on 7 February 2021. Retrieved 8 February 2021.
0. "Download Python" (https://www.python.org/downloads/release/python-391). *Python.org*. Archived (https://web.archive.org/web/20201208045225/https://www.python.org/downloads/release/python-391/) from the original on 8 December 2020. Retrieved 13 December 2020.
0. "history [vmspython]" (https://www.vmspython.org/doku.php?id=history). *www.vmspython.org*. Archived (https://web.archive.org/web/20201202194743/https://www.vmspython.org/doku.php?id=history) from the original on 2 December 2020. Retrieved 4 December 2020.
0. "An Interview with Guido van Rossum" (http://www.oreilly.com/pub/a/oreilly/frank/rossum_1099.html). Oreilly.com. Archived (https://web.archive.org/web/20140716222652/http://oreilly.com/pub/a/oreilly/frank/rossum_1099.html) from the original on 16 July 2014. Retrieved 24 November 2008.
0. "Download Python for Other Platforms" (https://www.python.org/download/other/). *Python.org*. Archived (https://web.archive.org/web/20201127015815/https://www.python.org/download/other/) from the original on 27 November 2020. Retrieved 4 December 2020.
0. "PyPy compatibility" (https://pypy.org/compat.html). Pypy.org. Archived (https://web.archive.org/web/20200606041845/https://www.pypy.org/compat.html) from the original on 6 June 2020. Retrieved 3 December 2012.
0. Team, The PyPy (28 December 2019). "Download and Install" (https://www.pypy.org/download.html). *PyPy*. Archived (https://web.archive.org/web/20220108212951/https://www.pypy.org/download.html) from the original on 8 January 2022. Retrieved 8 January 2022.

0. "speed comparison between CPython and Pypy" (https://speed.pypy.org/). Speed.pypy.org. Archived (https://web.archive.org/web/20210510014902/https://speed.pypy.org/) from the original on 10 May 2021. Retrieved 3 December 2012.
0. "Codon: Differences with Python" (https://docs.exaloop.io/codon/general/differences). Archived (https://web.archive.org/web/20230525002540/https://docs.exaloop.io/codon/general/differences) from the original on 25 May 2023. Retrieved 28 August 2023.
0. Lawson, Loraine (14 March 2023). "MIT-Created Compiler Speeds up Python Code" (https://thenewstack.io/mit-created-compiler-speeds-up-python-code/). *The New Stack*. Archived (https://web.archive.org/web/20230406054200/https://thenewstack.io/mit-created-compiler-speeds-up-python-code/) from the original on 6 April 2023. Retrieved 28 August 2023.
0. "Application-level Stackless features – PyPy 2.0.2 documentation" (http://doc.pypy.org/en/latest/stackless.html). Doc.pypy.org. Archived (https://web.archive.org/web/20200604231513/https://doc.pypy.org/en/latest/stackless.html) from the original on 4 June 2020. Retrieved 17 July 2013.
0. "Python-for-EV3" (https://education.lego.com/en-us/support/mindstorms-ev3/python-for-ev3). *LEGO Education*. Archived (https://web.archive.org/web/20200607234814/https://education.lego.com/en-us/support/mindstorms-ev3/python-for-ev3) from the original on 7 June 2020. Retrieved 17 April 2019.
0. Yegulalp, Serdar (29 October 2020). "Pyston returns from the dead to speed Python" (https://www.infoworld.com/article/3587591/pyston-returns-from-the-dead-to-speed-python.html). *InfoWorld*. Archived (https://web.archive.org/web/20210127113233/https://www.infoworld.com/article/3587591/pyston-returns-from-the-dead-to-speed-python.html) from the original on 27 January 2021. Retrieved 26 January 2021.
0. "cinder: Instagram's performance-oriented fork of CPython" (https://github.com/facebookincubator/cinder). *GitHub*. Archived (https://web.archive.org/web/20210504112500/https://github.com/facebookincubator/cinder) from the original on 4 May 2021. Retrieved 4 May 2021.
0. Aroca, Rafael (7 August 2021). "Snek Lang: feels like Python on Arduinos" (https://rafaelaroca.wordpress.com/2021/08/07/snek-lang-feels-like-python-on-arduinos/). *Yet Another Technology Blog*. Archived (https://web.archive.org/web/20240105001031/https://rafaelaroca.wordpress.com/2021/08/07/snek-lang-feels-like-python-on-arduinos/) from the original on 5 January 2024. Retrieved 4 January 2024.

0. Aufranc (CNXSoft), Jean-Luc (16 January 2020). "Snekboard Controls LEGO Power Functions with CircuitPython or Snek Programming Languages (Crowdfunding) – CNX Software" (https://www.cnx-software.com/2020/01/16/snekboard-controls-lego-power-functions-with-circuitpython-or-snek-programming-languages/). *CNX Software – Embedded Systems News*. Archived (https://web.archive.org/web/20240105001031/https://www.cnx-software.com/2020/01/16/snekboard-controls-lego-power-functions-with-circuitpython-or-snek-programming-languages/) from the original on 5 January 2024. Retrieved 4 January 2024.
0. Kennedy (@mkennedy), Michael. "Ready to find out if you're git famous?" (https://pythonbytes.fm/episodes/show/187/ready-to-find-out-if-youre-git-famous). *pythonbytes.fm*. Archived (https://web.archive.org/web/20240105001031/https://pythonbytes.fm/episodes/show/187/ready-to-find-out-if-youre-git-famous) from the original on 5 January 2024. Retrieved 4 January 2024.
0. Packard, Keith (20 December 2022). "The Snek Programming Language: A Python-inspired Embedded Computing Language" (https://sneklang.org/doc/snek.pdf) (PDF). Archived (https://web.archive.org/web/20240104162458/https://sneklang.org/doc/snek.pdf) (PDF) from the original on 4 January 2024. Retrieved 4 January 2024.
0. "Plans for optimizing Python" (https://code.google.com/p/unladen-swallow/wiki/ProjectPlan). *Google Project Hosting*. 15 December 2009. Archived (https://web.archive.org/web/20160411181848/https://code.google.com/p/unladen-swallow/wiki/ProjectPlan) from the original on 11 April 2016. Retrieved 24 September 2011.
0. "Python on the Nokia N900" (http://www.stochasticgeometry.ie/2010/04/29/python-on-the-nokia-n900/). *Stochastic Geometry*. 29 April 2010. Archived (https://web.archive.org/web/20190620000053/http://www.stochasticgeometry.ie/2010/04/29/python-on-the-nokia-n900/) from the original on 20 June 2019. Retrieved 9 July 2015.
0. "Brython" (https://brython.info/). *brython.info*. Archived (https://web.archive.org/web/20180803065954/http://brython.info/) from the original on 3 August 2018. Retrieved 21 January 2021.
0. "Transcrypt – Python in the browser" (https://www.transcrypt.org). *transcrypt.org*. Archived (https://web.archive.org/web/20180819133303/http://www.transcrypt.org/) from the original on 19 August 2018. Retrieved 22 December 2020.
0. "Transcrypt: Anatomy of a Python to JavaScript Compiler" (https://www.infoq.com/articles/transcrypt-python-javascript-compiler/). *InfoQ*. Archived (https://web.archive.org/web/20201205193339/https://www.infoq.com/articles/transcrypt-python-javascript-compiler/) from the original on 5 December 2020. Retrieved 20 January 2021.

0. "Nuitka Home | Nuitka Home" (http://nuitka.net/). *nuitka.net*. Archived (https://web.archive.org/web/20200530211233/https://nuitka.net/) from the original on 30 May 2020. Retrieved 18 August 2017.
0. Guelton, Serge; Brunet, Pierrick; Amini, Mehdi; Merlini, Adrien; Corbillon, Xavier; Raynaud, Alan (16 March 2015). "Pythran: enabling static optimization of scientific Python programs" (https://doi.org/10.1088%2F1749-4680%2F8%2F1%2F014001). *Computational Science & Discovery*. **8** (1). IOP Publishing: 014001. Bibcode:2015CS&D....8a4001G (https://ui.adsabs.harvard.edu/abs/2015CS&D....8a4001G). doi:10.1088/1749-4680/8/1/014001 (https://doi.org/10.1088%2F1749-4680%2F8%2F1%2F014001). ISSN 1749-4699 (https://search.worldcat.org/issn/1749-4699).
0. "The Python → 11l → C++ transpiler" (https://11l-lang.org/transpiler). Archived (https://web.archive.org/web/20220924233728/https://11l-lang.org/transpiler/) from the original on 24 September 2022. Retrieved 17 July 2022.
0. "google/grumpy" (https://github.com/google/grumpy). 10 April 2020. Archived (https://web.archive.org/web/20200415054919/https://github.com/google/grumpy) from the original on 15 April 2020. Retrieved 25 March 2020 – via GitHub.
0. "Projects" (https://opensource.google/projects/). *opensource.google*. Archived (https://web.archive.org/web/20200424191248/https://opensource.google/projects/) from the original on 24 April 2020. Retrieved 25 March 2020.
0. Francisco, Thomas Claburn in San. "Google's Grumpy code makes Python Go" (https://www.theregister.com/2017/01/05/googles_grumpy_makes_python_go/). *www.theregister.com*. Archived (https://web.archive.org/web/20210307165521/https://www.theregister.com/2017/01/05/googles_grumpy_makes_python_go/) from the original on 7 March 2021. Retrieved 20 January 2021.
0. "GitHub – IronLanguages/ironpython3: Implementation of Python 3.x for .NET Framework that is built on top of the Dynamic Language Runtime" (https://github.com/IronLanguages/ironpython3). *GitHub*. Archived (https://web.archive.org/web/20210928101250/https://github.com/IronLanguages/ironpython3) from the original on 28 September 2021.
0. "IronPython.net /" (https://ironpython.net/). *ironpython.net*. Archived (https://web.archive.org/web/20210417064418/https://ironpython.net/) from the original on 17 April 2021.
0. "Jython FAQ" (https://www.jython.org/jython-old-sites/archive/22/userfaq.html). *www.jython.org*. Archived (https://web.archive.org/web/20210422055726/https://www.jython.org/jython-old-sites/archive/22/userfaq.html) from the original on 22 April 2021. Retrieved 22 April 2021.

0. Murri, Riccardo (2013). *Performance of Python runtimes on a non-numeric scientific code*. European Conference on Python in Science (EuroSciPy). arXiv:1404.6388 (https://arxiv.org/abs/1404.6388). Bibcode:2014arXiv1404.6388M (https://ui.adsabs.harvard.edu/abs/2014arXiv1404.6388M).
0. "The Computer Language Benchmarks Game" (https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/python.html). Archived (https://web.archive.org/web/20200614210246/https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/python.html) from the original on 14 June 2020. Retrieved 30 April 2020.
0. Warsaw, Barry; Hylton, Jeremy; Goodger, David (13 June 2000). "PEP 1 – PEP Purpose and Guidelines" (https://www.python.org/dev/peps/pep-0001/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200606042011/https://www.python.org/dev/peps/pep-0001/) from the original on 6 June 2020. Retrieved 19 April 2011.
0. "PEP 8 – Style Guide for Python Code" (https://www.python.org/dev/peps/pep-0008/). *Python.org*. Archived (https://web.archive.org/web/20190417223549/https://www.python.org/dev/peps/pep-0008/) from the original on 17 April 2019. Retrieved 26 March 2019.
0. Cannon, Brett. "Guido, Some Guys, and a Mailing List: How Python is Developed" (https://web.archive.org/web/20090601134342/http://www.python.org/dev/intro/). *python.org*. Python Software Foundation. Archived from the original (https://www.python.org/dev/intro/) on 1 June 2009. Retrieved 27 June 2009.
0. "Moving Python's bugs to GitHub [LWN.net]" (https://lwn.net/Articles/885854/). Archived (https://web.archive.org/web/20221002183818/https://lwn.net/Articles/885854/) from the original on 2 October 2022. Retrieved 2 October 2022.
0. "Python Developer's Guide – Python Developer's Guide" (https://devguide.python.org/). *devguide.python.org*. Archived (https://web.archive.org/web/20201109032501/https://devguide.python.org/) from the original on 9 November 2020. Retrieved 17 December 2019.
0. Hughes, Owen (24 May 2021). "Programming languages: Why Python 4.0 might never arrive, according to its creator" (https://www.techrepublic.com/article/programming-languages-why-python-4-0-will-probably-never-arrive-according-to-its-creator/). *TechRepublic*. Archived (https://web.archive.org/web/20220714201302/https://www.techrepublic.com/article/programming-languages-why-python-4-0-will-probably-never-arrive-according-to-its-creator/) from the original on 14 July 2022. Retrieved 16 May 2022.
0. "PEP 602 – Annual Release Cycle for Python" (https://www.python.org/dev/peps/pep-0602/). *Python.org*. Archived (https://web.archive.org/web/20200614202755/https://www.python.org/dev/peps/pep-0602/) from the original on 14 June 2020. Retrieved 6 November 2019.

0. "Changing the Python release cadence [LWN.net]" (https://lwn.net/Articles/802777/). *lwn.net*. Archived (https://web.archive.org/web/20191106170153/https://lwn.net/Articles/802777/) from the original on 6 November 2019. Retrieved 6 November 2019.
0. Norwitz, Neal (8 April 2002). "[Python-Dev] Release Schedules (was Stability & change)" (https://mail.python.org/pipermail/python-dev/2002-April/022739.html). Archived (https://web.archive.org/web/20181215122750/https://mail.python.org/pipermail/python-dev/2002-April/022739.html) from the original on 15 December 2018. Retrieved 27 June 2009.
0. Aahz; Baxter, Anthony (15 March 2001). "PEP 6 – Bug Fix Releases" (https://www.python.org/dev/peps/pep-0006/). *Python Enhancement Proposals*. Python Software Foundation. Archived (https://web.archive.org/web/20200605001318/https://www.python.org/dev/peps/pep-0006/) from the original on 5 June 2020. Retrieved 27 June 2009.
0. "Python Buildbot" (https://www.python.org/dev/buildbot/). *Python Developer's Guide*. Python Software Foundation. Archived (https://web.archive.org/web/20200605001322/https://www.python.org/dev/buildbot/) from the original on 5 June 2020. Retrieved 24 September 2011.
0. "1. Extending Python with C or C++ – Python 3.9.1 documentation" (https://docs.python.org/3/extending/extending.html). *docs.python.org*. Archived (https://web.archive.org/web/20200623232830/https://docs.python.org/3/extending/extending.html) from the original on 23 June 2020. Retrieved 14 February 2021.
0. "PEP 623 – Remove wstr from Unicode" (https://www.python.org/dev/peps/pep-0623/). *Python.org*. Archived (https://web.archive.org/web/20210305153214/https://www.python.org/dev/peps/pep-0623/) from the original on 5 March 2021. Retrieved 14 February 2021.
0. "PEP 634 – Structural Pattern Matching: Specification" (https://www.python.org/dev/peps/pep-0634/). *Python.org*. Archived (https://web.archive.org/web/20210506005315/https://www.python.org/dev/peps/pep-0634/) from the original on 6 May 2021. Retrieved 14 February 2021.
0. "Documentation Tools" (https://wiki.python.org/moin/DocumentationTools). *Python.org*. Archived (https://web.archive.org/web/20201111173635/https://wiki.python.org/moin/DocumentationTools) from the original on 11 November 2020. Retrieved 22 March 2021.
0. "Whetting Your Appetite" (https://docs.python.org/tutorial/appetite.html). *The Python Tutorial*. Python Software Foundation. Archived (https://web.archive.org/web/20121026063559/http://docs.python.org/tutorial/appetite.html) from the original on 26 October 2012. Retrieved 20 February 2012.

0. "In Python, should I use else after a return in an if block?" (https://stackoverflow.com/questions/5033906/in-python-should-i-use-else-after-a-return-in-an-if-block). *Stack Overflow*. Stack Exchange. 17 February 2011. Archived (https://web.archive.org/web/20190620000050/https://stackoverflow.com/questions/5033906/in-python-should-i-use-else-after-a-return-in-an-if-block) from the original on 20 June 2019. Retrieved 6 May 2011.
0. Lutz, Mark (2009). *Learning Python: Powerful Object-Oriented Programming* (https://books.google.com/books?id=1HxWGezDZcgC&pg=PA17). O'Reilly Media, Inc. p. 17. ISBN 9781449379322. Archived (https://web.archive.org/web/20170717044012/https://books.google.com/books?id=1HxWGezDZcgC&pg=PA17) from the original on 17 July 2017. Retrieved 9 May 2017.
0. Fehily, Chris (2002). *Python* (https://books.google.com/books?id=carqdIdfVlYC&pg=PR15). Peachpit Press. p. xv. ISBN 9780201748840. Archived (https://web.archive.org/web/20170717044040/https://books.google.com/books?id=carqdIdfVlYC&pg=PR15) from the original on 17 July 2017. Retrieved 9 May 2017.
0. Lubanovic, Bill (2014). *Introducing Python* (http://archive.org/details/introducingpytho0000luba). Sebastopol, CA : O'Reilly Media. p. 305. ISBN 978-1-4493-5936-2. Retrieved 31 July 2023.
0. Blake, Troy (18 January 2021). "TIOBE Index for January 2021" (https://seniordba.wordpress.com/2021/01/18/tiobe-index-for-january-2021/). *Technology News and Information by SeniorDBA*. Archived (https://web.archive.org/web/20210321143253/https://seniordba.wordpress.com/2021/01/18/tiobe-index-for-january-2021/) from the original on 21 March 2021. Retrieved 26 February 2021.
0. "TIOBE Index" (https://www.tiobe.com/tiobe-index/). *TIOBE*. Retrieved 31 March 2025.
0. "Quotes about Python" (https://www.python.org/about/quotes/). Python Software Foundation. Archived (https://web.archive.org/web/20200603135201/https://www.python.org/about/quotes/) from the original on 3 June 2020. Retrieved 8 January 2012.
0. "Organizations Using Python" (https://wiki.python.org/moin/OrganizationsUsingPython). Python Software Foundation. Archived (https://web.archive.org/web/20180821075931/https://wiki.python.org/moin/OrganizationsUsingPython) from the original on 21 August 2018. Retrieved 15 January 2009.
0. "Python : the holy grail of programming" (http://cdsweb.cern.ch/journal/CERNBulletin/2006/31/News%20Articles/974627?ln=en). *CERN Bulletin* (31/2006). CERN Publications. 31 July 2006. Archived (https://archive.today/20130115191843/http://cdsweb.cern.ch/journal/CERNBulletin/2006/31/News%20Articles/974627?ln=en) from the original on 15 January 2013. Retrieved 11 February 2012.

0. Shafer, Daniel G. (17 January 2003). "Python Streamlines Space Shuttle Mission Design" (https://www.python.org/about/success/usa/). Python Software Foundation. Archived (https://web.archive.org/web/20200605093424/https://www.python.org/about/success/usa/) from the original on 5 June 2020. Retrieved 24 November 2008.
0. "Tornado: Facebook's Real-Time Web Framework for Python – Facebook for Developers" (https://developers.facebook.com/blog/post/301). *Facebook for Developers*. Archived (https://web.archive.org/web/20190219031313/https://developers.facebook.com/blog/post/301) from the original on 19 February 2019. Retrieved 19 June 2018.
0. "What Powers Instagram: Hundreds of Instances, Dozens of Technologies" (https://instagram-engineering.com/what-powers-instagram-hundreds-of-instances-dozens-of-technologies-adf2e22da2ad). Instagram Engineering. 11 December 2016. Archived (https://web.archive.org/web/20200615183410/https://instagram-engineering.com/what-powers-instagram-hundreds-of-instances-dozens-of-technologies-adf2e22da2ad) from the original on 15 June 2020. Retrieved 27 May 2019.
0. "How we use Python at Spotify" (https://labs.spotify.com/2013/03/20/how-we-use-python-at-spotify/). *Spotify Labs*. 20 March 2013. Archived (https://web.archive.org/web/20200610005143/https://labs.spotify.com/2013/03/20/how-we-use-python-at-spotify/) from the original on 10 June 2020. Retrieved 25 July 2018.
0. Fortenberry, Tim (17 January 2003). "Industrial Light & Magic Runs on Python" (https://www.python.org/about/success/ilm/). Python Software Foundation. Archived (https://web.archive.org/web/20200606042020/https://www.python.org/about/success/ilm/) from the original on 6 June 2020. Retrieved 11 February 2012.
0. Taft, Darryl K. (5 March 2007). "Python Slithers into Systems" (http://www.eweek.com/c/a/Application-Development/Python-Slithers-into-Systems/). *eWeek.com*. Ziff Davis Holdings. Archived (https://web.archive.org/web/20210813194304/https://www.eweek.com/development/python-slithers-into-systems/) from the original on 13 August 2021. Retrieved 24 September 2011.
0. *GitHub – reddit-archive/reddit: historical code from reddit.com.* (https://github.com/reddit-archive/reddit), The Reddit Archives, archived (https://web.archive.org/web/20200601104939/https://github.com/reddit-archive/reddit) from the original on 1 June 2020, retrieved 20 March 2019
0. "Real time communication at scale with Elixir at Discord" (https://elixir-lang.org/blog/2020/10/08/real-time-communication-at-scale-with-elixir-at-discord/). 8 October 2020.

0. "What Programming Language is Baidu Built In?" (https://www.freelancinggig.com/blog/2018/07/05/what-programming-language-is-baidu-built-in/#:~:text=Even%20though%20Baidu%20has%20used,part%20JavaScript%20has%20been%20applied). 5 July 2018.
0. "Usage statistics and market share of Python for websites" (http://w3techs.com/technologies/details/pl-python/all/all). 2012. Archived (https://web.archive.org/web/20210813194305/https://w3techs.com/technologies/details/pl-python) from the original on 13 August 2021. Retrieved 18 December 2012.
0. Oliphant, Travis (2007). "Python for Scientific Computing" (https://www.h2desk.com/blog/python-scientific-computing/). *Computing in Science and Engineering*. **9** (3): 10–20. Bibcode:2007CSE.....9c..10O (https://ui.adsabs.harvard.edu/abs/2007CSE.....9c..10O). CiteSeerX 10.1.1.474.6460 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.474.6460). doi:10.1109/MCSE.2007.58 (https://doi.org/10.1109%2FMCSE.2007.58). ISSN 1521-9615 (https://search.worldcat.org/issn/1521-9615). S2CID 206457124 (https://api.semanticscholar.org/CorpusID:206457124). Archived (https://web.archive.org/web/20200615193226/https://www.h2desk.com/blog/python-scientific-computing/) from the original on 15 June 2020. Retrieved 10 April 2015.
0. Millman, K. Jarrod; Aivazis, Michael (2011). "Python for Scientists and Engineers" (http://www.computer.org/csdl/mags/cs/2011/02/mcs2011020009.html). *Computing in Science and Engineering*. **13** (2): 9–12. Bibcode:2011CSE....13b...9M (https://ui.adsabs.harvard.edu/abs/2011CSE....13b...9M). doi:10.1109/MCSE.2011.36 (https://doi.org/10.1109%2FMCSE.2011.36). Archived (https://web.archive.org/web/20190219031439/https://www.computer.org/csdl/mags/cs/2011/02/mcs2011020009.html) from the original on 19 February 2019. Retrieved 7 July 2014.
0. *Science education with SageMath* (https://web.archive.org/web/20200615180428/http://visual.icse.us.edu.pl/methodology/why_Sage.html), Innovative Computing in Science Education, archived from the original (http://visual.icse.us.edu.pl/methodology/why_Sage.html) on 15 June 2020, retrieved 22 April 2019
0. "OpenCV: OpenCV-Python Tutorials" (https://docs.opencv.org/3.4.9/d6/d00/tutorial_py_root.html). *docs.opencv.org*. Archived (https://web.archive.org/web/20200923063145/https://docs.opencv.org/3.4.9/d6/d00/tutorial_py_root.html) from the original on 23 September 2020. Retrieved 14 September 2020.

0. Dean, Jeff; Monga, Rajat; et al. (9 November 2015). "TensorFlow: Large-scale machine learning on heterogeneous systems" (http://download.tensorflow.org/paper/whitepaper2015.pdf) (PDF). *TensorFlow.org*. Google Research. Archived (https://web.archive.org/web/20151120004649/http://download.tensorflow.org/paper/whitepaper2015.pdf) (PDF) from the original on 20 November 2015. Retrieved 10 November 2015.
0. Piatetsky, Gregory. "Python eats away at R: Top Software for Analytics, Data Science, Machine Learning in 2018: Trends and Analysis" (https://www.kdnuggets.com/2018/05/poll-tools-analytics-data-science-machine-learning-results.html/2). *KDnuggets*. Archived (https://web.archive.org/web/20191115234216/https://www.kdnuggets.com/2018/05/poll-tools-analytics-data-science-machine-learning-results.html/2) from the original on 15 November 2019. Retrieved 30 May 2018.
0. "Who is using scikit-learn? – scikit-learn 0.20.1 documentation" (https://scikit-learn.org/stable/testimonials/testimonials.html). *scikit-learn.org*. Archived (https://web.archive.org/web/20200506210716/https://scikit-learn.org/stable/testimonials/testimonials.html) from the original on 6 May 2020. Retrieved 30 November 2018.
0. Jouppi, Norm. "Google supercharges machine learning tasks with TPU custom chip" (https://cloudplatform.googleblog.com/2016/05/Google-supercharges-machine-learning-tasks-with-custom-chip.html). *Google Cloud Platform Blog*. Archived (https://web.archive.org/web/20160518201516/https://cloudplatform.googleblog.com/2016/05/Google-supercharges-machine-learning-tasks-with-custom-chip.html) from the original on 18 May 2016. Retrieved 19 May 2016.
0. De Raedt, Luc; Kimmig, Angelika (2015). "Probabilistic (logic) programming concepts" (https://doi.org/10.1007%2Fs10994-015-5494-z). *Machine Learning*. **100** (1): 5–47. doi:10.1007/s10994-015-5494-z (https://doi.org/10.1007%2Fs10994-015-5494-z). S2CID 3166992 (https://api.semanticscholar.org/CorpusID:3166992).
0. "Natural Language Toolkit – NLTK 3.5b1 documentation" (http://www.nltk.org/). *www.nltk.org*. Archived (https://web.archive.org/web/20200613003911/http://www.nltk.org/) from the original on 13 June 2020. Retrieved 10 April 2020.
0. Andersen, C. and Swift, T., 2023. The Janus System: a bridge to new prolog applications. In Prolog: The Next 50 Years (pp. 93–104). Cham: Springer Nature Switzerland.
0. "SWI-Prolog Python interface" (https://www.swi-prolog.org/pldoc/doc_for?object=section(%27packages/janus.html%27)). Archived (https://web.archive.org/web/20240315162046/https://www.swi-prolog.org/pldoc/doc_for?object=section%28%27packages%2Fjanus.html%27%29) from the original on 15 March 2024. Retrieved 15 March 2024.

0. Tarau, P., 2023. Reflections on automation, learnability and expressiveness in logic-based programming languages. In Prolog: The Next 50 Years (pp. 359–371). Cham: Springer Nature Switzerland.
0. "Tkinter — Python interface to TCL/Tk" (https://docs.python.org/3/library/tkinter.html). Archived (https://web.archive.org/web/20121018043136/http://docs.python.org/library/tkinter.html) from the original on 18 October 2012. Retrieved 9 June 2023.
0. "Python Tkinter Tutorial" (https://www.geeksforgeeks.org/python-tkinter-tutorial/). 3 June 2020. Archived (https://web.archive.org/web/20230609031631/https://www.geeksforgeeks.org/python-tkinter-tutorial/) from the original on 9 June 2023. Retrieved 9 June 2023.
0. "Installers for GIMP for Windows – Frequently Asked Questions" (https://web.archive.org/web/20130717070814/http://gimp-win.sourceforge.net/faq.html). 26 July 2013. Archived from the original (http://gimp-win.sourceforge.net/faq.html) on 17 July 2013. Retrieved 26 July 2013.
0. "jasc psp9components" (https://web.archive.org/web/20080319061519/http://www.jasc.com/support/customercare/articles/psp9components.asp). Archived from the original (http://www.jasc.com/support/customercare/articles/psp9components.asp) on 19 March 2008.
0. "About getting started with writing geoprocessing scripts" (http://webhelp.esri.com/arcgisdesktop/9.2/index.cfm?TopicName=About_getting_started_with_writing_geoprocessing_scripts). *ArcGIS Desktop Help 9.2*. Environmental Systems Research Institute. 17 November 2006. Archived (https://web.archive.org/web/20200605144616/http://webhelp.esri.com/arcgisdesktop/9.2/index.cfm?TopicName=About_getting_started_with_writing_geoprocessing_scripts) from the original on 5 June 2020. Retrieved 11 February 2012.
0. CCP porkbelly (24 August 2010). "Stackless Python 2.7" (https://community.eveonline.com/news/dev-blogs/stackless-python-2.7/). *EVE Community Dev Blogs*. CCP Games. Archived (https://web.archive.org/web/20140111155537/http://community.eveonline.com/news/dev-blogs/stackless-python-2.7/) from the original on 11 January 2014. Retrieved 11 January 2014. "As you may know, EVE has at its core the programming language known as Stackless Python."
0. Caudill, Barry (20 September 2005). "Modding Sid Meier's Civilization IV" (https://web.archive.org/web/20101202164144/http://www.2kgames.com/civ4/blog_03.htm). *Sid Meier's Civilization IV Developer Blog*. Firaxis Games. Archived from the original (http://www.2kgames.com/civ4/blog_03.htm) on 2 December 2010. "we created three levels of tools ... The next level offers Python and XML support, letting modders with more experience manipulate the game world and everything in it."

0. "Python Language Guide (v1.0)" (https://web.archive.org/web/20100715145616/http://code.google.com/apis/documents/docs/1.0/developers_guide_python.html). *Google Documents List Data API v1.0*. Archived from the original (https://code.google.com/apis/documents/docs/1.0/developers_guide_python.html) on 15 July 2010.
0. "Python Setup and Usage" (https://docs.python.org/3/using/unix.html). Python Software Foundation. Archived (https://web.archive.org/web/20200617143505/https://docs.python.org/3/using/unix.html) from the original on 17 June 2020. Retrieved 10 January 2020.
0. "Immunity: Knowing You're Secure" (https://web.archive.org/web/20090216134332/http://immunitysec.com/products-immdbg.shtml). Archived from the original on 16 February 2009.
0. "Core Security" (https://www.coresecurity.com/). *Core Security*. Archived (https://web.archive.org/web/20200609165041/http://www.coresecurity.com/) from the original on 9 June 2020. Retrieved 10 April 2020.
0. "What is Sugar?" (http://sugarlabs.org/go/Sugar). Sugar Labs. Archived (https://web.archive.org/web/20090109025944/http://sugarlabs.org/go/Sugar) from the original on 9 January 2009. Retrieved 11 February 2012.
0. "4.0 New Features and Fixes" (http://www.libreoffice.org/download/4-0-new-features-and-fixes/). *LibreOffice.org*. The Document Foundation. 2013. Archived (https://web.archive.org/web/20140209184807/http://www.libreoffice.org/download/4-0-new-features-and-fixes/) from the original on 9 February 2014. Retrieved 25 February 2013.
0. "Gotchas for Python Users" (https://web.archive.org/web/20081211062108/http://boo.codehaus.org/Gotchas+for+Python+Users). *boo.codehaus.org*. Codehaus Foundation. Archived from the original (http://boo.codehaus.org/Gotchas+for+Python+Users) on 11 December 2008. Retrieved 24 November 2008.
0. Esterbrook, Charles. "Acknowledgements" (https://web.archive.org/web/20080208141002/http://cobra-language.com/docs/acknowledgements/). *cobra-language.com*. Cobra Language. Archived from the original (http://cobra-language.com/docs/acknowledgements/) on 8 February 2008. Retrieved 7 April 2010.
0. "Proposals: iterators and generators [ES4 Wiki]" (https://web.archive.org/web/20071020082650/http://wiki.ecmascript.org/doku.php?id=proposals:iterators_and_generators). wiki.ecmascript.org. Archived from the original (http://wiki.ecmascript.org/doku.php?id=proposals:iterators_and_generators) on 20 October 2007. Retrieved 24 November 2008.

0. "Frequently asked questions" (https://docs.godotengine.org/en/stable/about/faq.html). *Godot Engine documentation*. Archived (https://web.archive.org/web/20210428053339/https://docs.godotengine.org/en/stable/about/faq.html) from the original on 28 April 2021. Retrieved 10 May 2021.
0. Kincaid, Jason (10 November 2009). "Google's Go: A New Programming Language That's Python Meets C++" (https://techcrunch.com/2009/11/10/google-go-language/). *TechCrunch*. Archived (https://web.archive.org/web/20100118014358/http://www.techcrunch.com/2009/11/10/google-go-language/) from the original on 18 January 2010. Retrieved 29 January 2010.
0. Strachan, James (29 August 2003). "Groovy – the birth of a new dynamic language for the Java platform" (https://web.archive.org/web/20070405085722/http://radio.weblogs.com/0112098/2003/08/29.html). Archived from the original (http://radio.weblogs.com/0112098/2003/08/29.html) on 5 April 2007. Retrieved 11 June 2007.
0. "Modular Docs – Why Mojo" (https://docs.modular.com/mojo/why-mojo.html). *docs.modular.com*. Archived (https://web.archive.org/web/20230505083518/https://docs.modular.com/mojo/why-mojo.html) from the original on 5 May 2023. Retrieved 5 May 2023. "Mojo as a member of the Python family [..] Embracing Python massively simplifies our design efforts, because most of the syntax is already specified. [..] we decided that the right long-term goal for Mojo is to provide a superset of Python (i.e. be compatible with existing programs) and to embrace the CPython immediately for long-tail ecosystem enablement. To a Python programmer, we expect and hope that Mojo will be immediately familiar, while also providing new tools for developing systems-level code that enable you to do things that Python falls back to C and C++ for."
0. Spencer, Michael (4 May 2023). "What is Mojo Programming Language?" (https://datasciencelearningcenter.substack.com/p/what-is-mojo-programming-language). *datasciencelearningcenter.substack.com*. Archived (https://web.archive.org/web/20230505090408/https://datasciencelearningcenter.substack.com/p/what-is-mojo-programming-language) from the original on 5 May 2023. Retrieved 5 May 2023.
0. Yegulalp, Serdar (16 January 2017). "Nim language draws from best of Python, Rust, Go, and Lisp" (https://www.infoworld.com/article/3157745/application-development/nim-language-draws-from-best-of-python-rust-go-and-lisp.html). *InfoWorld*. Archived (https://web.archive.org/web/20181013211847/https://www.infoworld.com/article/3157745/application-development/nim-language-draws-from-best-of-python-rust-go-and-lisp.html) from the original on 13 October 2018. Retrieved 7 June 2020. "Nim's syntax is strongly reminiscent of Python's, as it uses indented code blocks and some of the same syntax (such as the way if/elif/then/else blocks are constructed)."

0. "An Interview with the Creator of Ruby" (http://www.linuxdevcenter.com/pub/a/linux/2001/11/29/ruby.html). Linuxdevcenter.com. Archived (https://web.archive.org/web/20180428150410/http://www.linuxdevcenter.com/pub/a/linux/2001/11/29/ruby.html) from the original on 28 April 2018. Retrieved 3 December 2012.
0. Lattner, Chris (3 June 2014). "Chris Lattner's Homepage" (http://nondot.org/sabre). Chris Lattner. Archived (https://web.archive.org/web/20151222150510/http://nondot.org/sabre/) from the original on 22 December 2015. Retrieved 3 June 2014. "I started work on the Swift Programming Language in July of 2010. I implemented much of the basic language structure, with only a few people knowing of its existence. A few other (amazing) people started contributing in earnest late in 2011, and it became a major focus for the Apple Developer Tools group in July 2013 [...] drawing ideas from Objective-C, Rust, Haskell, Ruby, Python, C#, CLU, and far too many others to list."
0. Jalan, Nishant Aanjaney (10 November 2022). "Programming in Kotlin" (https://medium.com/codex/programming-in-kotlin-934bdb3659cf). *CodeX*. Retrieved 29 April 2024.
0. Kupries, Andreas; Fellows, Donal K. (14 September 2000). "TIP #3: TIP Format" (http://www.tcl.tk/cgi-bin/tct/tip/3.html). *tcl.tk*. Tcl Developer Xchange. Archived (https://web.archive.org/web/20170713233954/http://tcl.tk/cgi-bin/tct/tip/3.html) from the original on 13 July 2017. Retrieved 24 November 2008.
0. Gustafsson, Per; Niskanen, Raimo (29 January 2007). "EEP 1: EEP Purpose and Guidelines" (http://www.erlang.org/eeps/eep-0001.html). erlang.org. Archived (https://web.archive.org/web/20200615153206/http://erlang.org/eeps/eep-0001.html) from the original on 15 June 2020. Retrieved 19 April 2011.
0. "Swift Evolution Process" (https://github.com/apple/swift-evolution/blob/master/process.md). *Swift Programming Language Evolution repository on GitHub*. 18 February 2020. Archived (https://web.archive.org/web/20200427182556/https://github.com/apple/swift-evolution/blob/master/process.md) from the original on 27 April 2020. Retrieved 27 April 2020.

Sources

- "Python for Artificial Intelligence" (https://web.archive.org/web/20121101045354/http://wiki.python.org/moin/PythonForArtificialIntelligence). Python Wiki. 19 July 2012. Archived from the original (https://wiki.python.org/moin/PythonForArtificialIntelligence) on 1 November 2012. Retrieved 3 December 2012.
- Paine, Jocelyn, ed. (August 2005). "AI in Python" (https://web.archive.org/web/20120326105810/http://www.ainewsletter.com/newsletters/aix_0508.htm#python_ai_ai). *AI Expert Newsletter*. Amzi!. Archived from the original (http://www.ainewsletter.com/newsletters/aix_0508.htm#python_ai_ai) on 26 March 2012. Retrieved 11 February 2012.

- "PyAIML 0.8.5 : Python Package Index" (https://pypi.python.org/pypi/PyAIML). Pypi.python.org. Retrieved 17 July 2013.
- Russell, Stuart J. & Norvig, Peter (2009). *Artificial Intelligence: A Modern Approach* (3rd ed.). Upper Saddle River, NJ: Prentice Hall. ISBN 978-0-13-604259-4.

Further reading

- Downey, Allen (July 2024). *Think Python: How to Think Like a Computer Scientist* (https://allendowney.github.io/ThinkPython/) (3rd ed.). O'Reilly Media. ISBN 978-1098155438.
- Lutz, Mark (2013). *Learning Python* (5th ed.). O'Reilly Media. ISBN 978-0-596-15806-4.
- Summerfield, Mark (2009). *Programming in Python 3* (2nd ed.). Addison-Wesley Professional. ISBN 978-0-321-68056-3.
- Ramalho, Luciano (May 2022). *Fluent Python* (https://www.thoughtworks.com/insights/books/fluent-python-2nd-edition). O'Reilly Media. ISBN 978-1-4920-5632-4.

External links

- Official website (https://www.python.org/)
- The Python Tutorial (https://docs.python.org/3/tutorial/)

PDF

Portable Document Format

Adobe PDF icon

Filename extension	`.pdf`

Internet media type	`application/pdf`,[1]
	`application/x-pdf`
	`application/x-bzpdf`
	`application/x-gzpdf`

Type code	PDF [1] (including a single trailing space)
Uniform Type Identifier (UTI)	com.adobe.pdf
Magic number	%PDF
Developed by	Adobe Inc. (1991–2008) ISO (2008–)
Initial release	June 15, 1993
Latest release	2.0
Extended to	PDF/A, PDF/E, PDF/UA, PDF/VT, PDF/X
Standard	ISO 32000-2
Open format?	Yes
Website	iso.org/standard/75839.html (https://iso.org/standard/75839.html)

Portable Document Format (**PDF**), standardized as **ISO 32000**, is a file format developed by Adobe in 1992 to present documents, including text formatting and images, in a manner independent of application software, hardware, and operating systems.[2][3] Based on the PostScript language, each PDF file encapsulates a complete description of a fixed-layout flat document, including the text, fonts, vector graphics, raster images and other information needed to display it. PDF has its roots in "The Camelot Project" initiated by Adobe co-founder John Warnock in 1991.[4] PDF was standardized as ISO 32000 in 2008.[5] The last edition as ISO 32000-2:2020 was published in December 2020.[6]

PDF files may contain a variety of content besides flat text and graphics including logical structuring elements, interactive elements such as annotations and form-fields, layers, rich media (including video content), three-dimensional objects using U3D or PRC, and various other data formats. The PDF specification also provides for encryption and digital signatures, file attachments, and metadata to enable workflows requiring these features.

History

The development of PDF began in 1991 when John Warnock wrote a paper for a project then code-named Camelot, in which he proposed the creation of a simplified version of PostScript called Interchange PostScript (IPS).[7] Unlike traditional PostScript, which was tightly focused on rendering print jobs to output devices, IPS would be optimized for displaying pages to any screen and any platform.[7]

Adobe Systems made the PDF specification available free of charge in 1993. In the early years PDF was popular mainly in desktop publishing workflows, and competed with several other formats, including DjVu, Envoy, Common Ground Digital Paper, Farallon Replica and even Adobe's own PostScript format.

PDF was a proprietary format controlled by Adobe until it was released as an open standard on July 1, 2008, and published by the International Organization for Standardization as ISO 32000-1:2008,[8][9] at which time control of the specification passed to an ISO Committee of volunteer industry experts. In 2008, Adobe published a Public Patent License to ISO 32000-1 granting royalty-free rights for all patents owned by Adobe necessary to make, use, sell, and distribute PDF-compliant implementations.[10]

PDF 1.7, the sixth edition of the PDF specification that became ISO 32000-1, includes some proprietary technologies defined only by Adobe, such as Adobe XML Forms Architecture (XFA) and JavaScript extension for Acrobat, which are referenced by ISO 32000-1 as normative and indispensable for the full implementation of the ISO 32000-1 specification.[11] These proprietary technologies are not standardized, and their specification is published only on Adobe's website.[12][13][14] Many of them are not supported by popular third-party implementations of PDF.

ISO published version 2.0 of PDF, ISO 32000-2 in 2017, available for purchase, replacing the free specification provided by Adobe.[15] In December 2020, the second edition of PDF 2.0, ISO 32000-2:2020, was published, with clarifications, corrections, and critical updates to normative references[16] (ISO 32000-2 does not include any proprietary technologies as normative references).[17] In April 2023 the PDF Association made ISO 32000-2 available for download free of charge.[15]

Technical details

A PDF file is often a combination of vector graphics, text, and bitmap graphics. The basic types of content in a PDF are:

- Typeset text stored as content streams (i.e., not encoded in plain text);
- Vector graphics for illustrations and designs that consist of shapes and lines;

- Raster graphics for photographs and other types of images; and
- Other multimedia objects.

In later PDF revisions, a PDF document can also support links (inside document or web page), forms, JavaScript (initially available as a plugin for Acrobat 3.0), or any other types of embedded contents that can be handled using plug-ins.
PDF combines three technologies:

- An equivalent subset of the PostScript page description programming language but in declarative form, for generating the layout and graphics.
- A font-embedding/replacement system to allow fonts to travel with the documents.
- A structured storage system to bundle these elements and any associated content into a single file, with data compression where appropriate.

PostScript language

PostScript is a page description language run in an interpreter to generate an image.[7] It can handle graphics and has standard features of programming languages such as branching and looping.[7] PDF is a subset of PostScript, simplified to remove such control flow features, while graphics commands remain.[7]
PostScript was originally designed for a drastically different use case: transmission of one-way linear print jobs in which the PostScript interpreter would collect a series of commands until it encountered the `showpage` command, then execute all the commands to render a page as a raster image to a printing device.[18] PostScript was not intended for long-term storage and real-time interactive rendering of electronic documents to computer monitors, so there was no need to support anything other than consecutive rendering of pages.[18] If there was an error in the final printed output, the user would correct it at the application level and send a new print job in the form of an entirely new PostScript file. Thus, any given page in a PostScript file could be accurately rendered only as the cumulative result of executing all preceding commands to draw all previous pages—any of which could affect subsequent pages—plus the commands to draw that particular page, and there was no easy way to bypass that process to skip around to different pages.[18]
Traditionally, to go from PostScript to PDF, a source PostScript file (that is, an executable program) is used as the basis for generating PostScript-like PDF code (see, e.g., Adobe Distiller). This is done by applying standard compiler techniques like loop unrolling, inlining and removing unused branches, resulting in code that is purely declarative and static.[18] The result is then packaged into a container format, together with all necessary dependencies for correct rendering (external files, graphics, or fonts to which the document refers), and compressed. Modern applications write to printer drivers that directly generate PDF rather than going through PostScript first.
As a document format, PDF has several advantages over PostScript:

- PDF contains only static declarative PostScript code that can be processed as data, and does not require a full program interpreter or compiler.[18] This

avoids the complexity and security risks of an engine with such a higher complexity level.
- Like Display PostScript, PDF has supported transparent graphics since version 1.4, while standard PostScript does not.
- PDF enforces the rule that the code for any particular page cannot affect any other pages.[18] That rule is strongly recommended for PostScript code too, but has to be implemented explicitly (see, e.g., the Document Structuring Conventions), as PostScript is a full programming language that allows for such greater flexibilities and is not limited to the concepts of pages and documents.
- All data required for rendering is included within the file itself, improving portability.[19]

Its disadvantages are:

- A loss of flexibility, and limitation to a single use case.
- A (sometimes much) larger file size.[20]

PDF since v1.6 supports embedding of interactive 3D documents: 3D drawings can be embedded using U3D or PRC and various other data formats.[21][22][23]

File format

A PDF file is organized using ASCII characters, except for certain elements that may have binary content. The file starts with a header containing a magic number (as a readable string) and the version of the format, for example `%PDF-1.7`. The format is a subset of a COS ("Carousel" Object Structure) format.[24] A COS tree file consists primarily of *objects*, of which there are nine types:[17]

- Boolean values, representing *true* or *false*
- Real numbers
- Integers
- Strings, enclosed within parentheses (`(...)`) or represented as hexadecimal within single angle brackets (`<...>`). Strings may contain 8-bit characters.
- Names, starting with a forward slash (`/`)
- Arrays, ordered collections of objects enclosed within square brackets (`[...]`)
- Dictionaries, collections of objects indexed by names enclosed within double angle brackets (`<<...>>`)
- Streams, usually containing large amounts of optionally compressed binary data, preceded by a dictionary and enclosed between the `stream` and `endstream` keywords.
- The null object

Comments using 8-bit characters prefixed with the percent sign (`%`) may be inserted.

Objects may be either *direct* (embedded in another object) or *indirect*. Indirect objects are numbered with an *object number* and a *generation number* and defined between the `obj` and `endobj` keywords if residing in the document root. Beginning with PDF version 1.5, indirect objects (except other streams) may also be located in special streams known as *object streams* (marked `/Type /ObjStm`). This technique enables non-stream objects to have standard stream filters applied to them, reduces the size of files that have large numbers of small indirect objects and is especially useful for *Tagged PDF*. Object streams do not support specifying an object's *generation number* (other than 0).
An index table, also called the cross-reference table, is located near the end of the file and gives the byte offset of each indirect object from the start of the file.[25] This design allows for efficient random access to the objects in the file, and also allows for small changes to be made without rewriting the entire file (*incremental update*). Before PDF version 1.5, the table would always be in a special ASCII format, be marked with the `xref` keyword, and follow the main body composed of indirect objects. Version 1.5 introduced optional *cross-reference streams*, which have the form of a standard stream object, possibly with filters applied. Such a stream may be used instead of the ASCII cross-reference table and contains the offsets and other information in binary format. The format is flexible in that it allows for integer width specification (using the `/W` array), so that for example, a document not exceeding 64 KiB in size may dedicate only 2 bytes for object offsets.
At the end of a PDF file is a footer containing

- The `startxref` keyword followed by an offset to the start of the cross-reference table (starting with the `xref` keyword) or the cross-reference stream object, followed by
- The `%%EOF` end-of-file marker.

If a cross-reference stream is not being used, the footer is preceded by the `trailer` keyword followed by a dictionary containing information that would otherwise be contained in the cross-reference stream object's dictionary:

- A reference to the root object of the tree structure, also known as the *catalog* (`/Root`)
- The count of indirect objects in the cross-reference table (`/Size`)
- Other optional information

Within each page, there are one or multiple content streams that describe the text, vector and images being drawn on the page. The content stream is stack-based, similar to PostScript.[26]

There are two layouts to the PDF files: non-linearized (not "optimized") and linearized ("optimized"). Non-linearized PDF files can be smaller than their linear counterparts, though they are slower to access because portions of the data required to assemble pages of the document are scattered throughout the PDF file. Linearized PDF files (also called "optimized" or "web optimized" PDF files) are constructed in a manner that enables them to be read in a Web browser plugin without waiting for the entire file to download, since all objects required for the first page to display are optimally organized at the start of the file.[27] PDF files may be optimized using Adobe Acrobat software or QPDF.

The maximum size of an Acrobat PDF page, superimposed on a map of Europe.

Page dimensions are not limited by the format itself. However, Adobe Acrobat imposes a limit of 15 million by 15 million inches, or 225 trillion in^2 (145,161 km^2).[2]:1129

Imaging model

The basic design of how graphics are represented in PDF is very similar to that of PostScript, except for the use of transparency, which was added in PDF 1.4.

PDF graphics use a device-independent Cartesian coordinate system to describe the surface of a page. A PDF page description can use a matrix to scale, rotate, or skew graphical elements. A key concept in PDF is that of the *graphics state*, which is a collection of graphical parameters that may be changed, saved, and restored by a *page description*. PDF has (as of version 2.0) 25 graphics state properties, of which some of the most important are:

- The *current transformation matrix* (CTM), which determines the coordinate system
- The *clipping path*
- The *color space*
- The *alpha constant*, which is a key component of transparency
- *Black point compensation* control (introduced in PDF 2.0)

Vector graphics

As in PostScript, vector graphics in PDF are constructed with *paths*. Paths are usually composed of lines and cubic Bézier curves, but can also be constructed from the outlines of text. Unlike PostScript, PDF does not allow a single path to mix text outlines with lines and curves. Paths can be stroked, filled, fill then stroked, or used for clipping. Strokes and fills can use any color set in the graphics state, including *patterns*. PDF supports several types of patterns. The simplest is the *tiling pattern* in which a piece of artwork is specified to be drawn repeatedly. This may be a *colored tiling pattern*, with the colors specified in the pattern object, or an *uncolored tiling pattern*, which defers color specification to the time the pattern is drawn. Beginning with PDF 1.3

there is also a *shading pattern*, which draws continuously varying colors. There are seven types of shading patterns of which the simplest are the *axial shading* (Type 2) and *radial shading* (Type 3).

Raster images

Raster images in PDF (called *Image XObjects*) are represented by dictionaries with an associated stream. The dictionary describes the properties of the image, and the stream contains the image data. (Less commonly, small raster images may be embedded directly in a page description as an *inline image*.) Images are typically *filtered* for compression purposes. Image filters supported in PDF include the following general-purpose filters:

- *ASCII85Decode*, a filter used to put the stream into 7-bit ASCII,
- *ASCIIHexDecode*, similar to ASCII85Decode but less compact,
- *FlateDecode*, a commonly used filter based on the deflate algorithm defined in RFC 1951 (https://datatracker.ietf.org/doc/html/rfc1951) (deflate is also used in the gzip, PNG, and zip file formats among others); introduced in PDF 1.2; it can use one of two groups of predictor functions for more compact zlib/deflate compression: *Predictor 2* from the TIFF 6.0 specification and predictors (filters) from the PNG specification (RFC 2083 (https://datatracker.ietf.org/doc/html/rfc2083)),
- *LZWDecode*, a filter based on LZW Compression; it can use one of two groups of predictor functions for more compact LZW compression: *Predictor 2* from the TIFF 6.0 specification and predictors (filters) from the PNG specification,
- *RunLengthDecode*, a simple compression method for streams with repetitive data using the run-length encoding algorithm and the image-specific filters,
- *DCTDecode*, a lossy filter based on the JPEG standard,
- *CCITTFaxDecode*, a lossless bi-level (black/white) filter based on the Group 3 or Group 4 CCITT (ITU-T) fax compression standard defined in ITU-T T.4 and T.6,
- *JBIG2Decode*, a lossy or lossless bi-level (black/white) filter based on the JBIG2 standard, introduced in PDF 1.4, and
- *JPXDecode*, a lossy or lossless filter based on the JPEG 2000 standard, introduced in PDF 1.5.

Normally all image content in a PDF is embedded in the file. But PDF allows image data to be stored in external files by the use of *external streams* or *Alternate Images*. Standardized subsets of PDF, including PDF/A and PDF/X, prohibit these features.

Text

Text in PDF is represented by *text elements* in page content streams. A text element specifies that *characters* should be drawn at certain positions. The characters are specified using the *encoding* of a selected *font resource*.

A font object in PDF is a description of a digital typeface. It may either describe the characteristics of a typeface, or it may include an embedded *font file*. The latter case is called an *embedded font* while the former is called an *unembedded font*. The font files that may be embedded are based on widely used standard digital font formats: Type 1 (and its compressed variant CFF), TrueType, and (beginning with PDF 1.6) OpenType. Additionally PDF supports the Type 3 variant in which the components of the font are described by PDF graphic operators.
Fourteen typefaces, known as the *standard 14 fonts*, have a special significance in PDF documents:

- Times (v3) (in regular, italic, bold, and bold italic)
- Courier (in regular, oblique, bold and bold oblique)
- Helvetica (v3) (in regular, oblique, bold and bold oblique)
- Symbol
- Zapf Dingbats

These fonts are sometimes called the *base fourteen fonts*.[28] These fonts, or suitable substitute fonts with the same metrics, should be available in most PDF readers, but they are not *guaranteed* to be available in the reader, and may only display correctly if the system has them installed.[29] Fonts may be substituted if they are not embedded in a PDF.
Within text strings, characters are shown using *character codes* (integers) that map to glyphs in the current font using an *encoding*. There are several predefined encodings, including *WinAnsi*, *MacRoman*, and many encodings for East Asian languages and a font can have its own built-in encoding. (Although the WinAnsi and MacRoman encodings are derived from the historical properties of the Windows and Macintosh operating systems, fonts using these encodings work equally well on any platform.) PDF can specify a predefined encoding to use, the font's built-in encoding or provide a lookup table of differences to a predefined or built-in encoding (not recommended with TrueType fonts).[2] The encoding mechanisms in PDF were designed for Type 1 fonts, and the rules for applying them to TrueType fonts are complex.
For large fonts or fonts with non-standard glyphs, the special encodings *Identity-H* (for horizontal writing) and *Identity-V* (for vertical) are used. With such fonts, it is necessary to provide a *ToUnicode* table if semantic information about the characters is to be preserved.
A text document which is scanned to PDF without the text being recognised by optical character recognition (OCR) is an image, with no fonts or text properties.

Transparency

The original imaging model of PDF was *opaque,* similar to PostScript, where each object drawn on the page completely replaced anything previously marked in the same location. In PDF 1.4 the imaging model was extended to allow transparency. When transparency is used, new objects interact with previously marked objects to produce blending effects. The addition of transparency to PDF was done by means of new extensions that were designed to be ignored in products written to PDF 1.3 and earlier specifications. As a result, files that use a small amount of transparency might be viewed acceptably by older viewers, but files making extensive use of transparency could be viewed incorrectly by an older viewer.

The transparency extensions are based on the key concepts of *transparency groups*, *blending modes*, *shape*, and *alpha*. The model is closely aligned with the features of Adobe Illustrator version 9. The blend modes were based on those used by Adobe Photoshop at the time. When the PDF 1.4 specification was published, the formulas for calculating blend modes were kept secret by Adobe. They have since been published.[30]
The concept of a transparency group in PDF specification is independent of existing notions of "group" or "layer" in applications such as Adobe Illustrator. Those groupings reflect logical relationships among objects that are meaningful when editing those objects, but they are not part of the imaging model.

Additional features

Logical structure and accessibility

A **tagged PDF** (see clause 14.8 in ISO 32000) includes document structure and semantics information to enable reliable text extraction and accessibility.[31] Technically speaking, tagged PDF is a stylized use of the format that builds on the logical structure framework introduced in PDF 1.3. Tagged PDF defines a set of standard structure types and attributes that allow page content (text, graphics, and images) to be extracted and reused for other purposes.[32]
Tagged PDF is not required in situations where a PDF file is intended only for print. Since the feature is optional, and since the rules for tagged PDF were relatively vague in ISO 32000-1, support for tagged PDF among consuming devices, including assistive technology (AT), is uneven as of 2021.[33] ISO 32000-2, however, includes an improved discussion of tagged PDF which is anticipated to facilitate further adoption.
An ISO-standardized subset of PDF specifically targeted at accessibility, PDF/UA, was first published in 2012.

Optional Content Groups (layers)

With the introduction of PDF version 1.5 (2003) came the concept of Layers. Layers, more formally known as Optional Content Groups (OCGs), refer to sections of content in a PDF document that can be selectively viewed or hidden by document authors or viewers. This capability is useful in CAD drawings, layered artwork, maps, multi-language documents, etc.
Basically, it consists of an Optional Content Properties Dictionary added to the document root. This dictionary contains an array of Optional Content Groups (OCGs), each describing a set of information and each of which may be individually displayed or suppressed, plus a set of Optional Content Configuration Dictionaries, which give the status (Displayed or Suppressed) of the given OCGs.

Encryption and signatures

A PDF file may be encrypted, for security, in which case a password is needed to view or edit the contents. PDF 2.0 defines 256-bit AES encryption as the standard for PDF 2.0 files. The PDF Reference also defines ways that third parties can define their own encryption systems for PDF.

PDF files may be digitally signed, to provide secure authentication; complete details on implementing digital signatures in PDF are provided in ISO 32000-2.
PDF files may also contain embedded DRM restrictions that provide further controls that limit copying, editing, or printing. These restrictions depend on the reader software to obey them, so the security they provide is limited.
The standard security provided by PDF consists of two different methods and two different passwords: a *user password*, which encrypts the file and prevents opening, and an *owner password*, which specifies operations that should be restricted even when the document is decrypted, which can include modifying, printing, or copying text and graphics out of the document, or adding or modifying text notes and AcroForm fields. The user password encrypts the file, while the owner password does not, instead relying on client software to respect these restrictions. An owner password can easily be removed by software, including some free online services.[34] Thus, the use restrictions that a document author places on a PDF document are not secure, and cannot be assured once the file is distributed; this warning is displayed when applying such restrictions using Adobe Acrobat software to create or edit PDF files.
Even without removing the password, most freeware or open source PDF readers ignore the permission "protections" and allow the user to print or make copies of excerpts of the text as if the document were not limited by password protection.[35][36][37]
Beginning with PDF 1.5, Usage rights (UR) signatures are used to enable additional interactive features that are not available by default in a particular PDF viewer application. The signature is used to validate that the permissions have been granted by a bona fide granting authority. For example, it can be used to allow a user:[38]

- To save the PDF document along with a modified form or annotation data
- Import form data files in FDF, XFDF, and text (CSV/TSV) formats
- Export form data files in FDF and XFDF formats
- Submit form data
- Instantiate new pages from named page templates
- Apply a digital signature to existing digital signature form field
- Create, delete, modify, copy, import, and export annotations

For example, Adobe Systems grants permissions to enable additional features in Adobe Reader, using public-key cryptography. Adobe Reader verifies that the signature uses a certificate from an Adobe-authorized certificate authority. Any PDF application can use this same mechanism for its own purposes.[38]
Under specific circumstances including non-patched systems of the receiver, the information the receiver of a digital signed document sees can be manipulated by the sender after the document has been signed by the signer.[39]
PAdES (*PDF Advanced Electronic Signatures*) is a set of restrictions and extensions to PDF and ISO 32000-1[40] making it suitable for advanced electronic signatures. This is published by ETSI as TS 102 778.[41]

File attachments

PDF files can have file attachments which processors may access and open or save to a local filesystem.[42]

Metadata

PDF files can contain two types of metadata.[2] The first is the Document Information Dictionary, a set of key/value fields such as author, title, subject, creation and update dates. This is optional and is referenced from an `Info` key in the trailer of the file. A small set of fields is defined and can be extended with additional text values if required. This method is deprecated in PDF 2.0.

In PDF 1.4, support was added for Metadata Streams, using the Extensible Metadata Platform (XMP) to add XML standards-based extensible metadata as used in other file formats. PDF 2.0 allows metadata to be attached to any object in the document, such as information about embedded illustrations, fonts, and images, as well as the whole document (attaching to the document catalog), using an extensible schema.

PDF documents can also contain display settings, including the page display layout and zoom level in a Viewer Preferences object. Adobe Reader uses these settings to override the user's default settings when opening the document.[43] The free Adobe Reader cannot remove these settings.

Accessibility

PDF files can be created specifically to be accessible to people with disabilities.[44][45][46][47][48] PDF file formats in use as of 2014 can include tags, text equivalents, captions, audio descriptions, and more. Some software can automatically produce tagged PDFs, but this feature is not always enabled by default.[49][50] Leading screen readers, including JAWS, Window-Eyes, Hal, and Kurzweil 1000 and 3000 can read tagged PDFs.[51][52] Moreover, tagged PDFs can be re-flowed and magnified for readers with visual impairments. Adding tags to older PDFs and those that are generated from scanned documents can present some challenges.

One of the significant challenges with PDF accessibility is that PDF documents have three distinct views, which, depending on the document's creation, can be inconsistent with each other. The three views are (i) the physical view, (ii) the tags view, and (iii) the content view. The physical view is displayed and printed (what most people consider a PDF document). The tags view is what screen readers and other assistive technologies use to deliver high-quality navigation and reading experience to users with disabilities. The content view is based on the physical order of objects within the PDF's content stream and may be displayed by software that does not fully support the tags' view, such as the Reflow feature in Adobe's Reader.

PDF/UA, the International Standard for accessible PDF based on ISO 32000-1 was first published as ISO 14289–1 in 2012 and establishes normative language for accessible PDF technology.

Multimedia

Rich Media PDF is a PDF file including interactive content that can be embedded or linked within the file. It can contain images, audio, video content, or buttons. For example, if the interactive PDF is a digital catalog for an E-commerce business, products can be listed on the PDF pages and can be added with images and links to the website and buttons to order directly from the document.

Forms

Interactive Forms is a mechanism to add forms to the PDF file format. PDF currently supports two different methods for integrating data and PDF forms. Both formats today coexist in the PDF specification:[38][53][54][55]

- AcroForms (also known as Acrobat forms), introduced in the PDF 1.2 format specification and included in all later PDF specifications.
- XML Forms Architecture (XFA) forms, introduced in the PDF 1.5 format specification. Adobe XFA Forms are not compatible with AcroForms.[56] XFA was deprecated from PDF with PDF 2.0.

AcroForms were introduced in the PDF 1.2 format. AcroForms permit the uses of objects (*e.g.* text boxes, Radio buttons, *etc.*) and some code (*e.g.* JavaScript). Alongside the standard PDF action types, interactive forms (AcroForms) support submitting, resetting, and importing data. The "submit" action transmits the names and values of selected interactive form fields to a specified uniform resource locator (URL). Interactive form field names and values may be submitted in any of the following formats, (depending on the settings of the action's ExportFormat, SubmitPDF, and XFDF flags):[38]

HTML Form format
HTML 4.01 Specification since PDF 1.5; HTML 2.0 since 1.2
Forms Data Format (FDF)
based on PDF, uses the same syntax and has essentially the same file structure, but is much simpler than PDF since the body of an FDF document consists of only one required object. Forms Data Format is defined in the PDF specification (since PDF 1.2). The Forms Data Format can be used when submitting form data to a server, receiving the response, and incorporating it into the interactive form. It can also be used to export form data to stand-alone files that can be imported back into the corresponding PDF interactive form. FDF was originally defined in 1996 as part of ISO 32000-2:2017.
XML Forms Data Format (XFDF)
(external XML Forms Data Format Specification, Version 2.0; supported since PDF 1.5; it replaced the "XML" form submission format defined in PDF 1.4) the XML version of Forms Data Format, but the XFDF implements only a subset of FDF containing forms and annotations. Some entries in the FDF dictionary do not have XFDF equivalents – such as the Status, Encoding, JavaScript, Page's keys, EmbeddedFDFs, Differences, and Target. In addition, XFDF does not allow the spawning, or addition, of new pages based on the given data; as

can be done when using an FDF file. The XFDF specification is referenced (but not included) in PDF 1.5 specification (and in later versions). It is described separately in *XML Forms Data Format Specification*.[57] The PDF 1.4 specification allowed form submissions in XML format, but this was replaced by submissions in XFDF format in the PDF 1.5 specification. XFDF conforms to the XML standard. XFDF can be used in the same way as FDF; e.g., form data is submitted to a server, modifications are made, then sent back and the new form data is imported in an interactive form. It can also be used to export form data to stand-alone files that can be imported back into the corresponding PDF interactive form. As of August 2019, XFDF 3.0 is an ISO/IEC standard under the formal name *ISO 19444-1:2019 - Document management — XML Forms Data Format — Part 1: Use of ISO 32000-2 (XFDF 3.0)*.[58] This standard is a normative reference of ISO 32000-2.

PDF

The entire document can be submitted rather than individual fields and values, as was defined in PDF 1.4.

AcroForms can keep form field values in external stand-alone files containing key-value pairs. The external files may use Forms Data Format (FDF) and XML Forms Data Format (XFDF) files.[59][57][60] The usage rights (UR) signatures define rights for import form data files in FDF, XFDF, and text (CSV/TSV) formats, and export form data files in FDF and XFDF formats.[38]

In PDF 1.5, Adobe Systems introduced a proprietary format for forms; Adobe XML Forms Architecture (XFA). Adobe XFA Forms are not compatible with ISO 32000's AcroForms feature, and most PDF processors do not handle XFA content. The XFA specification is referenced from ISO 32000-1/PDF 1.7 as an external proprietary specification and was entirely deprecated from PDF with ISO 32000-2 (PDF 2.0).

Licensing

Anyone may create applications that can read and write PDF files without having to pay royalties to Adobe Systems; Adobe holds patents to PDF, but licenses them for royalty-free use in developing software complying with its PDF specification.[61]

Security

Changes to content

In November 2019, researchers from Ruhr University Bochum and Hackmanit GmbH published attacks on digitally signed PDFs.[62] They showed how to change the visible content in a signed PDF without invalidating the signature in 21 of 22 desktop PDF viewers and 6 of 8 online validation services by abusing implementation flaws. At the same conference, they additionally showed how to exfiltrate the plaintext of encrypted content in PDFs.[63] In 2021, they showed new so-called *shadow attacks* on PDFs that abuse the flexibility of features provided in the

specification.[64] An overview of security issues in PDFs regarding denial of service, information disclosure, data manipulation, and arbitrary code execution attacks was presented by Jens Müller.[65][66]

Malware vulnerability

PDF files can be infected with viruses, Trojans, and other malware. They can have hidden JavaScript code that might exploit vulnerabilities in a PDF, hidden objects executed when the file that hides them is opened, and, less commonly, a malicious PDF can launch malware.[67]

PDF attachments carrying viruses were first discovered in 2001. The virus, named *OUTLOOK.PDFWorm* or *Peachy*, uses Microsoft Outlook to send itself as an attached Adobe PDF file. It was activated with Adobe Acrobat, but not with Acrobat Reader.[68]

From time to time, new vulnerabilities are discovered in various versions of Adobe Reader,[69] prompting the company to issue security fixes. Other PDF readers are also susceptible. One aggravating factor is that a PDF reader can be configured to start automatically if a web page has an embedded PDF file, providing a vector for attack. If a malicious web page contains an infected PDF file that takes advantage of a vulnerability in the PDF reader, the system may be compromised even if the browser is secure. Some of these vulnerabilities are a result of the PDF standard allowing PDF documents to be scripted with JavaScript. Disabling JavaScript execution in the PDF reader can help mitigate such future exploits, although it does not protect against exploits in other parts of the PDF viewing software. Security experts say that JavaScript is not essential for a PDF reader and that the security benefit that comes from disabling JavaScript outweighs any compatibility issues caused.[70] One way of avoiding PDF file exploits is to have a local or web service convert files to another format before viewing.

On March 30, 2010, security researcher Didier Stevens reported an Adobe Reader and Foxit Reader exploit that runs a malicious executable if the user allows it to launch when asked.[71]

Software

Viewers and editors

Many PDF viewers are provided free of charge from a variety of sources. Programs to manipulate and edit PDF files are available, usually for purchase.

There are many software options for creating PDFs, including the PDF printing capabilities built into macOS, iOS,[72] and most Linux distributions. Much document processing software including LibreOffice, Microsoft Office 2007 (if updated to SP2) and later,[73] WordPerfect 9, and Scribus can export documents in PDF. There are many PDF print drivers for Microsoft Windows, the pdfTeX typesetting system, the DocBook PDF tools, applications developed around Ghostscript and Adobe Acrobat itself as well as Adobe InDesign, Adobe FrameMaker, Adobe Illustrator, Adobe Photoshop, that allow a "PDF printer" to be set up, which when selected sends output to a PDF file instead of a physical printer. Google's online office suite Google Docs allows uploading and saving to PDF. Some web apps offer free PDF editing and annotation tools.

The Free Software Foundation was "developing a free, high-quality and fully functional set of libraries and programs that implement the PDF file format and associated technologies to the ISO 32000 standard", as one of its high priority projects.[74][75] In 2011, however, the GNU PDF project was removed from the list of "high priority projects" due to the maturation of the Poppler library,[76] which has enjoyed wider use in applications such as Evince with the GNOME desktop environment. Poppler is based on Xpdf[77][78] code base. There are also commercial development libraries available as listed in List of PDF software.
The Apache PDFBox project of the Apache Software Foundation is an open source Java library, licensed under the Apache License, for working with PDF documents.[79]

Printing

Raster image processors (RIPs) are used to convert PDF files into a raster format suitable for imaging onto paper and other media in printers, digital production presses and prepress in a process known as rasterization. RIPs capable of processing PDF directly include the Adobe PDF Print Engine[80] from Adobe Systems and Jaws[81] and the Harlequin RIP from Global Graphics.
In 1993, the Jaws raster image processor from Global Graphics became the first shipping prepress RIP that interpreted PDF natively without conversion to another format. The company released an upgrade to its Harlequin RIP with the same capability in 1997.[82]
Agfa-Gevaert introduced and shipped Apogee, the first prepress workflow system based on PDF, in 1997.
Many commercial offset printers have accepted the submission of press-ready PDF files as a print source, specifically the PDF/X-1a subset and variations of the same.[83] The submission of press-ready PDF files is a replacement for the problematic need for receiving collected native working files.
In 2006, PDF was widely accepted as the standard print job format at the Open Source Development Labs Printing Summit. It is supported as a print job format by the Common Unix Printing System and desktop application projects such as GNOME, KDE, Firefox, Thunderbird, LibreOffice and OpenOffice have switched to emit print jobs in PDF.[84]
Some desktop printers also support direct PDF printing, which can interpret PDF data without external help.

Native display model

PDF was selected as the "native" metafile format for macOS (originally called Mac OS X), replacing the PICT format of the earlier classic Mac OS. The imaging model of the Quartz graphics layer is based on the model common to Display PostScript and PDF, leading to the nickname *Display PDF*. The Preview application can display PDF files, as can version 2.0 and later of the Safari web browser. System-level support for PDF allows macOS applications to create PDF documents automatically, provided they support the OS-standard printing architecture. The files are then exported in PDF 1.3 format according to the file header. When taking a screenshot under Mac OS X versions 10.0 through 10.3, the image was also captured as a PDF; later versions save screen captures as a PNG file, though this behavior can be set back to PDF if desired.

Annotation

Adobe Acrobat is one example of proprietary software that allows the user to annotate, highlight, and add notes to already created PDF files. One UNIX application available as free software (under the GNU General Public License) is PDFedit. The freeware Foxit Reader, available for Microsoft Windows, macOS and Linux, allows annotating documents. Tracker Software's PDF-XChange Viewer allows annotations and markups without restrictions in its freeware alternative. Apple's macOS's integrated PDF viewer, Preview, does also enable annotations as does the open-source software Skim, with the latter supporting interaction with LaTeX, SyncTeX, and PDFSync and integration with BibDesk reference management software. Freeware Qiqqa can create an annotation report that summarizes all the annotations and notes one has made across their library of PDFs. The Text Verification Tool exports differences in documents as annotations and markups.

There are also web annotation systems that support annotation in pdf and other document formats. In cases where PDFs are expected to have all of the functionality of paper documents, ink annotation is required.

Conversion and Information Extraction

PDF's emphasis on preserving the visual appearance of documents across different software and hardware platforms poses challenges to the conversion of PDF documents to other file formats and the targeted extraction of information, such as text, images, tables, bibliographic information, and document metadata. Numerous tools and source code libraries support these tasks. Several labeled datasets to test PDF conversion and information extraction tools exist and have been used for benchmark evaluations of the tool's performance.[85]

Alternatives

The Open XML Paper Specification is a competing format used both as a page description language and as the native print spooler format for Microsoft Windows since Windows Vista.

Mixed Object: Document Content Architecture is a competing format. MO:DCA-P is a part of Advanced Function Presentation.

See also

- ebook
- Web page
- XSL Formatting Objects
- Page margin
- PDF portfolio

References

0. Hardy, M.; Masinter, L.; Markovic, D.; Johnson, D.; Bailey, M. (March 2017). *The application/pdf Media Type* (https://datatracker.ietf.org/doc/html/rfc8118). IETF. doi:10.17487/RFC8118 (https://doi.org/10.17487%2FRFC8118). RFC 8118 (https://datatracker.ietf.org/doc/html/rfc8118).
0. Adobe Systems Incorporated (November 2006). "PDF Reference" (https://web.archive.org/web/20081001170454/https://www.adobe.com/devnet/acrobat/pdfs/pdf_reference_1-7.pdf) (PDF). 1.7 (6th ed.). Archived from the original (https://www.adobe.com/devnet/acrobat/pdfs/pdf_reference_1-7.pdf) (PDF) on October 1, 2008. Retrieved January 12, 2023.
0. Warnock, J. (October 14, 2004) [Original date 5 May 1995]. "The Camelot Project" (https://www.pdfa.org/norm-refs/warnock_camelot.pdf) (PDF). Archived (https://web.archive.org/web/20110718230852/http://www.planetpdf.com/planetpdf/pdfs/warnock_camelot.pdf) (PDF) from the original on July 18, 2011.
0. "What is a PDF? Portable Document Format | Adobe Acrobat DC" (https://www.adobe.com/acrobat/about-adobe-pdf.html). Adobe Systems Inc. Archived (https://web.archive.org/web/20230130032548/https://www.adobe.com/acrobat/about-adobe-pdf.html) from the original on January 30, 2023. Retrieved January 12, 2023.
0. "ISO 32000-1:2008" (https://web.archive.org/web/20180726064724/http://wwwimages.adobe.com/www.adobe.com/content/dam/acom/en/devnet/pdf/pdfs/PDF32000_2008.pdf) (PDF). Archived from the original (http://wwwimages.adobe.com/www.adobe.com/content/dam/acom/en/devnet/pdf/pdfs/PDF32000_2008.pdf) (PDF) on July 26, 2018.
0. "ISO 32000-2 – PDF Association" (https://pdfa.org/resource/iso-32000-2/). Retrieved January 27, 2025.
0. Pfiffner, Pamela (2003). *Inside the Publishing Revolution: The Adobe Story*. Berkeley: Peachpit Press. p. 137. ISBN 0-321-11564-3.
0. "ISO 32000-1:2008 – Document management – Portable document format – Part 1: PDF 1.7" (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51502). ISO. July 1, 2008. Archived (https://web.archive.org/web/20101206175751/http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51502) from the original on December 6, 2010. Retrieved February 21, 2010.
0. Orion, Egan (December 5, 2007). "PDF 1.7 is approved as ISO 32000" (https://web.archive.org/web/20071213004627/http://www.theinquirer.net/gb/inquirer/news/2007/12/05/pdf-approved-iso-32000). *The Inquirer*. Archived from the original (http://www.theinquirer.net/gb/inquirer/news/2007/12/05/pdf-approved-iso-32000) on December 13, 2007. Retrieved December 5, 2007.

0. "Public Patent License, ISO 32000-1: 2008 – PDF 1.7" (https://www.adobe.com/pdf/pdfs/ISO32000-1PublicPatentLicense.pdf) (PDF). Adobe Systems Inc. 2008. Archived (https://web.archive.org/web/20090618144613/http://www.adobe.com/pdf/pdfs/ISO32000-1PublicPatentLicense.pdf) (PDF) from the original on June 18, 2009. Retrieved January 12, 2023.
0. "Guide for the procurement of standards-based ICT – Elements of Good Practice, Against lock-in: building open ICT systems by making better use of standards in public procurement" (https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=SWD:2013:0224:FIN:EN:PDF). European Commission. June 25, 2013. Archived (https://web.archive.org/web/20200919174545/https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=SWD:2013:0224:FIN:EN:PDF) from the original on September 19, 2020. Retrieved January 12, 2023. "Example: ISO/IEC 29500, ISO/IEC 26300 and ISO 32000 for document formats reference information that is not accessible by all parties (references to proprietary technology and brand names, incomplete scope or dead web links)."
0. "ISO/TC 171/SC 2/WG 8 N 603 – Meeting Report" (https://web.archive.org/web/20121126013025/http://pdf.editme.com/files/pdfREF-meetings/ISO-TC171-SC2-WG8_N0603_SC2WG8_MtgRept_SLC.pdf) (PDF). *Edit me*. June 27, 2011. Archived from the original (http://pdf.editme.com/files/pdfREF-meetings/ISO-TC171-SC2-WG8_N0603_SC2WG8_MtgRept_SLC.pdf) (PDF) on November 26, 2012 – via Archive. "XFA is not to be ISO standard just yet. The Committee urges Adobe Systems to submit the XFA Specification, XML Forms Architecture (XFA), to ISO for standardization The Committee is concerned about the stability of the XFA specification Part 2 will reference XFA 3.1"
0. "Embedding and publishing interactive, 3-dimensional, scientific figures in Portable Document Format (PDF) files" (https://doi.org/10.1371%2Fjournal.pone.0069446.s001). *PLOS ONE*. **8** (9). 2013. doi:10.1371/journal.pone.0069446.s001 (https://doi.org/10.1371%2Fjournal.pone.0069446.s001). "the implementation of the U3D standard was not complete and proprietary extensions were used."
0. Rosenthol, Leonard (2012). "PDF and Standards" (https://web.archive.org/web/20130902000323/http://cdn.parleys.com/p/5148922a0364bc17fc56c6e5/iSUM2012_00_LRO_presentation.pdf) (PDF). Adobe Systems. Archived from the original (http://cdn.parleys.com/p/5148922a0364bc17fc56c6e5/iSUM2012_00_LRO_presentation.pdf) (PDF) on September 2, 2013. Retrieved October 20, 2013 – via Parleys.
0. "Announcing no-cost access to the latest PDF standard: ISO 32000-2 (PDF 2.0)" (https://pdfa.org/sponsored-standards) (Press release). PDF Association. June 16, 2023 [Updated; originally published 5 April 2023]. Archived (https://web.archive.org/web/20230923202322/https://pdfa.org/sponsored-standards/) from the original on September 23, 2023. Retrieved October 6, 2023.

0. "ISO 32000-2:2020 is now available" (https://www.pdfa.org/iso-32000-22020-is-now-available/). PDFA. December 14, 2020. Archived (https://web.archive.org/web/20221204112238/https://www.pdfa.org/iso-32000-22020-is-now-available/) from the original on December 4, 2022. Retrieved February 3, 2021.
0. "ISO 32000-2 – Document management — Portable document format — Part 2: PDF 2.0" (https://www.iso.org/standard/75839.html). ISO. January 5, 2021. Archived (https://web.archive.org/web/20210128003836/https://www.iso.org/standard/75839.html) from the original on January 28, 2021. Retrieved February 3, 2021.
0. Pfiffner, Pamela (2003). *Inside the Publishing Revolution: The Adobe Story*. Berkeley: Peachpit Press. p. 139. ISBN 0-321-11564-3.
0. "PostScript Language Reference" (https://web.archive.org/web/20210724120635/https://www.adobe.com/content/dam/acom/en/devnet/actionscript/articles/PLRM.pdf) (PDF). Archived from the original (https://www.adobe.com/content/dam/acom/en/devnet/actionscript/articles/PLRM.pdf) (PDF) on July 24, 2021.
0. Anton Ertl, Martin. "What is the PDF format good for?" (https://www.complang.tuwien.ac.at/anton/why-not-pdf.html). *complang.tuwien.ac.at*. Vienna University of Technology. Archived (https://web.archive.org/web/20240404031526/https://www.complang.tuwien.ac.at/anton/why-not-pdf.html) from the original on April 4, 2024. Retrieved April 8, 2024.
0. "3D supported formats" (https://web.archive.org/web/20100212072951/http://www.adobe.com/manufacturing/resources/3dformats/). Adobe Systems Inc. July 14, 2009. Archived from the original (https://www.adobe.com/manufacturing/resources/3dformats/) on February 12, 2010. Retrieved February 21, 2010.
0. "Supported file formats in Acrobat and Reader" (https://helpx.adobe.com/acrobat/kb/supported-file-formats-acrobat-reader.html#main_2D_and_3D_formats__Acrobat_9_Pro_Extended__Adobe_3D_Reviewer_). Adobe Systems Inc. November 11, 2022. Archived (https://web.archive.org/web/20221221111958/https://helpx.adobe.com/acrobat/kb/supported-file-formats-acrobat-reader.html#main_2D_and_3D_formats__Acrobat_9_Pro_Extended__Adobe_3D_Reviewer_) from the original on December 21, 2022. Retrieved January 12, 2023.
0. "JavaScript for Acrobat 3D | Adobe Acrobat Developer Center" (https://web.archive.org/web/20091112231130/https://www.adobe.com/devnet/acrobat/javascript_acrobt_3d.html). Adobe Systems Inc. Archived from the original (https://www.adobe.com/devnet/acrobat/javascript_acrobt_3d.html) on November 12, 2009. Retrieved January 12, 2023.
0. Pravetz, Jim. "In Defense of COS, or Why I Love JSON and Hate XML" (https://web.archive.org/web/20140502013429/http://jimpravetz.com/blog/2012/12/in-defense-of-cos/). *jimpravetz.com*. Archived from the original on May 2, 2014.

0. Adobe Systems, PDF Reference, pp. 39–40.
0. PikePdf documentation. "Working with content streams" (https://pikepdf.readthedocs.io/en/latest/topics/content_streams.html). Archived (https://web.archive.org/web/20220705084446/https://pikepdf.readthedocs.io/en/latest/topics/content_streams.html) from the original on July 5, 2022. Retrieved May 8, 2022.
0. "Adobe Developer Connection: PDF Reference and Adobe Extensions to the PDF Specification" (https://web.archive.org/web/20061115132507/https://www.adobe.com/devnet/pdf/pdf_reference.html). Adobe Systems Inc. Archived from the original (https://www.adobe.com/devnet/pdf/pdf_reference.html) on November 15, 2006. Retrieved December 13, 2010.
0. Howard, Jacci. "Desktop Publishing: Base 14 Fonts – Definition" (https://web.archive.org/web/20160614134144/http://desktoppub.about.com/od/glossary/g/base14fonts.htm). *About.com Tech*. Archived from the original (http://desktoppub.about.com/od/glossary/g/base14fonts.htm) on June 14, 2016.
0. Merz, Thomas (June 2003). "The PDF Font Aquarium" (https://web.archive.org/web/20110718231502/http://www.planetpdf.com/planetpdf/pdfs/pdf2k/03e/merz_fontaquarium.pdf) (PDF). Archived from the original on July 18, 2011.
0. "PDF Blend Modes Addendum" (https://web.archive.org/web/20111014100004/https://www.adobe.com/content/dam/Adobe/en/devnet/pdf/pdfs/pdf_reference_archives/blend_modes.pdf) (PDF). Archived from the original (https://www.adobe.com/content/dam/Adobe/en/devnet/pdf/pdfs/pdf_reference_archives/blend_modes.pdf) (PDF) on October 14, 2011. Retrieved January 12, 2023.
0. "Tagged PDF Best Practice Guide: Syntax" (https://pdfa.org/wp-content/uploads/2019/06/TaggedPDFBestPracticeGuideSyntax.pdf) (PDF). *pdfa.org*. PDF Association. June 2019. Retrieved June 24, 2024.
0. Johnson, Duff (April 22, 2004). "What is Tagged PDF?" (https://www.talkingpdf.org/what-is-tagged-pdf/). Archived (https://web.archive.org/web/20040807132851/http://www.planetpdf.com/enterprise/article.asp?ContentID=6067) from the original on August 7, 2004.
0. "Is PDF accessible?" (https://www.washington.edu/doit/pdf-accessible?1002=). *DO-IT - Disabilities, Opportunities, Internetworking, and Technology*. University of Washington. October 4, 2022. Archived (https://web.archive.org/web/20230210114239/https://www.washington.edu/doit/pdf-accessible?1002=) from the original on February 10, 2023. Retrieved January 12, 2023.
0. "FreeMyPDF.com – Removes passwords from viewable PDFs" (http://freemypdf.com/). *freemypdf.com*. Archived (https://web.archive.org/web/20210220014724/https://freemypdf.com/) from the original on February 20, 2021. Retrieved June 23, 2009.

0. Kirk, Jeremy (December 4, 2008). "Adobe admits new PDF password protection is weaker" (http://www.macworld.com/article/1137343/pdf.html). *Macworld*. IDG Communications Inc. Archived (https://web.archive.org/web/20170117225255/http://www.macworld.com/article/1137343/pdf.html) from the original on January 17, 2017. Retrieved September 14, 2016.
0. Guignard, Bryan. "How secure is PDF" (https://web.archive.org/web/20051024235303/https://www.cs.cmu.edu/~dst/Adobe/Gallery/PDFsecurity.pdf) (PDF). Carnegie Mellon University. Archived from the original (https://www.cs.cmu.edu/~dst/Adobe/Gallery/PDFsecurity.pdf) (PDF) on October 24, 2005.
0. Merz, Thomas (November 2001). *PDF Security Overview: Strengths and Weaknesses* (https://web.archive.org/web/20101011050457/http://www.planetpdf.com/planetpdf/pdfs/pdf2k/01W/merz_securitykeynote.pdf) (PDF). PDF 2001 conference. Scottsdale/Arizona. Archived from the original on October 11, 2010.
0. Adobe Systems Inc. (July 1, 2008). "Document Management – Portable Document Format – Part 1: PDF 1.7, First Edition" (https://web.archive.org/web/20081203002256/https://www.adobe.com/devnet/acrobat/pdfs/PDF32000_2008.pdf) (PDF). Archived from the original (https://www.adobe.com/devnet/acrobat/pdfs/PDF32000_2008.pdf) (PDF) on December 3, 2008. Retrieved January 12, 2023.
0. "PDF Insecurity Website" (https://pdf-insecurity.org/signature-shadow/shadow-attacks.html). *pdf-insecurity.org*. Archived (https://web.archive.org/web/20230326024850/https://pdf-insecurity.org/signature-shadow/shadow-attacks.html) from the original on March 26, 2023. Retrieved January 12, 2023.
0. "ISO 32000-1:2008 Document management -- Portable document format -- Part 1: PDF 1.7" (http://www.iso.org/iso/catalogue_detail.htm?csnumber=51502). International Organization for Standardization ISO. Archived (https://web.archive.org/web/20170210072051/http://www.iso.org/iso/catalogue_detail.htm?csnumber=51502) from the original on February 10, 2017. Retrieved March 22, 2016.
0. "ETSI TS 102 778-1 - Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 1: PAdES Overview - a framework document for PAdES" (http://www.etsi.org/deliver/etsi_ts%5C102700_102799%5C10277801%5C01.01.01_60%5Cts_10277801v010101p.pdf) (PDF). 1.1.1. European Telecommunications Standards Institute ETSI. July 2009. Archived (https://web.archive.org/web/20230308052536/https://www.etsi.org/deliver/etsi_ts%5C102700_102799%5C10277801%5C01.01.01_60%5Cts_10277801v010101p.pdf) (PDF) from the original on March 8, 2023. Retrieved January 12, 2023.

0. "Links and attachments in PDFs" (https://helpx.adobe.com/uk/acrobat/using/links-attachments-pdfs.html). Archived (https://web.archive.org/web/20210423155713/https://helpx.adobe.com/uk/acrobat/using/links-attachments-pdfs.html) from the original on April 23, 2021. Retrieved April 23, 2021.
0. "Getting Familiar with Adobe Reader > Understanding Preferences" (http://www.adobepress.com/articles/article.asp?p=412914). *Adobe Press*. Pearson. September 2, 2005. Archived (https://web.archive.org/web/20121023144614/https://www.adobepress.com/articles/article.asp?p=412914) from the original on October 23, 2012. Retrieved January 12, 2023.
0. "PDF Accessibility" (https://webaim.org/techniques/acrobat/). WebAIM. Archived (https://web.archive.org/web/20230112153237/https://webaim.org/techniques/acrobat/) from the original on January 12, 2023. Retrieved January 12, 2023.
0. Clark, Joe (August 22, 2005). "Facts and Opinions About PDF Accessibility" (http://www.alistapart.com/articles/pdf_accessibility). Archived (https://web.archive.org/web/20130124140051/http://alistapart.com/articles/pdf_accessibility) from the original on January 24, 2013. Retrieved January 12, 2023.
0. "Accessibility and PDF documents" (https://web.archive.org/web/20100427062242/http://wac.osu.edu/pdf/). *Web Accessibility Center*. The Ohio State University. Archived from the original (http://wac.osu.edu/pdf/) on April 27, 2010. Retrieved January 12, 2023.
0. "PDF Accessibility Standards" (https://web.archive.org/web/20100529035503/http://www.bbc.co.uk/guidelines/futuremedia/accessibility/accessible_pdf.shtml). 1.2. BBC. Archived from the original (https://www.bbc.co.uk/guidelines/futuremedia/accessibility/accessible_pdf.shtml) on May 29, 2010. Retrieved January 12, 2023.
0. "PDF Accessibility" (https://web.archive.org/web/20100527215445/http://www.csus.edu/training/handouts/workshops/creating_accessible_pdfs.pdf) (PDF). California State University. 2009. Archived from the original (http://www.csus.edu/training/handouts/workshops/creating_accessible_pdfs.pdf) (PDF) on May 27, 2010. Retrieved January 12, 2023.
0. "LibreOffice Help – Export as PDF" (https://help.libreoffice.org/7.4/en-US/text/shared/01/ref_pdf_export.html). Archived (https://web.archive.org/web/20230112153247/https://help.libreoffice.org/7.4/en-US/text/shared/01/ref_pdf_export.html) from the original on January 12, 2023. Retrieved January 12, 2023.

0. Z., Andrew (January 11, 2008). "Exporting PDF/A for long-term archiving" (http://www.ooninja.com/2008/01/generating-pdfa-for-long-term-archiving.html). Archived (https://web.archive.org/web/20210224185200/https://www.ooninja.com/2008/01/generating-pdfa-for-long-term-archiving.html) from the original on February 24, 2021. Retrieved September 22, 2012.
0. Biersdorfer, J.D. (April 10, 2009). "Tip of the Week: Adobe Reader's 'Read Aloud' Feature" (https://gadgetwise.blogs.nytimes.com/2009/04/10/tip-of-the-week-adobe-readers-read-aloud-feature/). *The New York Times*. Archived (https://web.archive.org/web/20201122205912/https://gadgetwise.blogs.nytimes.com/2009/04/10/tip-of-the-week-adobe-readers-read-aloud-feature/) from the original on November 22, 2020. Retrieved January 12, 2023.
0. "Accessing PDF documents with assistive technology: A screen reader user's guide" (https://web.archive.org/web/20080728093103/https://www.adobe.com/accessibility/pdfs/accessing-pdf-sr.pdf) (PDF). Adobe Systems Inc. Archived from the original (https://www.adobe.com/accessibility/pdfs/accessing-pdf-sr.pdf) (PDF) on July 28, 2008. Retrieved January 12, 2023.
0. "Gnu PDF – PDF Knowledge – Forms Data Format" (https://web.archive.org/web/20130101054615/http://www.gnupdf.org/Forms_Data_Format). Archived from the original on January 1, 2013. Retrieved January 12, 2023.
0. "About PDF forms" (https://web.archive.org/web/20110429032948/http://livedocs.adobe.com/coldfusion/8/htmldocs/help.html?content=formsPDF_02.html). Adobe Systems Inc. Archived from the original (http://livedocs.adobe.com/coldfusion/8/htmldocs/help.html?content=formsPDF_02.html) on April 29, 2011. Retrieved February 19, 2010.
0. Demling, Peter (July 1, 2008). "Convert XFA Form to AcroForm?" (https://community.adobe.com/t5/acrobat-sdk-discussions/convert-xfa-form-to-acroform/td-p/1175365). Archived (https://web.archive.org/web/20230112153241/https://community.adobe.com/t5/acrobat-sdk-discussions/convert-xfa-form-to-acroform/td-p/1175365) from the original on January 12, 2023. Retrieved January 12, 2023.
0. "Migrating from Adobe Acrobat forms to XML forms" (https://web.archive.org/web/20101006151011/http://partners.adobe.com/public/developer/tips/topic_tip2.html). Archived from the original (http://partners.adobe.com/public/developer/tips/topic_tip2.html) on October 6, 2010. Retrieved January 12, 2023.
0. "XML Forms Data Format Specification, version 2" (https://web.archive.org/web/20180730100811/https://www.immagic.com/eLibrary/ARCHIVES/TECH/ADOBE/A070914X.pdf) (PDF). September 2007. Archived from the original (https://www.immagic.com/eLibrary/ARCHIVES/TECH/ADOBE/A070914X.pdf) (PDF) on July 30, 2018. Retrieved February 19, 2010.

0. "ISO 19444-1:2019(en)" (https://www.iso.org/obp/ui/#iso:std:iso:19444:-1:ed-2:v1:en). The International Organization for Standardization. Archived (https://web.archive.org/web/20160617031837/https://www.iso.org/obp/ui/#iso:std:iso:19444:-1:ed-2:v1:en) from the original on June 17, 2016. Retrieved December 3, 2020.
0. Adobe Systems Incorporated (September 20, 2022). "Using Acrobat forms and form data on the web" (https://helpx.adobe.com/acrobat/kb/acrobat-forms-form-data-web.html). Archived (https://web.archive.org/web/20230112153235/https://helpx.adobe.com/acrobat/kb/acrobat-forms-form-data-web.html) from the original on January 12, 2023. Retrieved January 12, 2023.
0. "FDF Data Exchange Specification" (https://web.archive.org/web/20081203041943/https://www.adobe.com/devnet/acrobat/pdfs/fdf_data_exchange.pdf) (PDF). February 8, 2007. Archived from the original (https://www.adobe.com/devnet/acrobat/pdfs/fdf_data_exchange.pdf) (PDF) on December 3, 2008. Retrieved January 12, 2023.
0. "Developer Resources" (https://web.archive.org/web/20160227041714/http://partners.adobe.com/public/developer/support/topic_legal_notices.html). Adobe Systems Inc. Archived from the original (http://partners.adobe.com/public/developer/support/topic_legal_notices.html) on February 27, 2016.
0. Mladenov, Vladislav; Mainka, Christian; Meyer Zu Selhausen, Karsten; Grothe, Martin; Schwenk, Jörg (November 6, 2019). "1 Trillion Dollar Refund: How to Spoof PDF Signatures". *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (https://dl.acm.org/doi/10.1145/3319535.3339812). CCS '19. ACM Digital Library, ACM SIGSAC Conference on Computer and Communications Security. pp. 1–14. doi:10.1145/3319535.3339812 (https://doi.org/10.1145%2F3319535.3339812). ISBN 9781450367479. S2CID 199367545 (https://api.semanticscholar.org/CorpusID:199367545). Archived (https://web.archive.org/web/20210426223722/https://dl.acm.org/doi/10.1145/3319535.3339812) from the original on April 26, 2021. Retrieved April 6, 2021.
0. Müller, Jens; Ising, Fabian; Mladenov, Vladislav; Mainka, Christian; Schinzel, Sebastian; Schwenk, Jörg (November 6, 2019). "Practical Decryption exFiltration: Breaking PDF Encryption". *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (https://dl.acm.org/doi/10.1145/3319535.3354214). CCS '19. ACM Digital Library, ACM SIGSAC Conference on Computer and Communications Security. pp. 15–29. doi:10.1145/3319535.3354214 (https://doi.org/10.1145%2F3319535.3354214). ISBN 9781450367479. S2CID 207959243 (https://api.semanticscholar.org/CorpusID:207959243). Archived (https://web.archive.org/web/20210426223415/https://dl.acm.org/doi/10.1145/3319535.3354214) from the original on April 26, 2021. Retrieved April 6, 2021.

0. "Shadow Attacks: Hiding and Replacing Content in Signed PDFs" (https://www.ndss-symposium.org/ndss-paper/shadow-attacks-hiding-and-replacing-content-in-signed-pdfs/). Internet Society, The Network and Distributed System Security Symposium. Archived (https://web.archive.org/web/20210421094100/https://www.ndss-symposium.org/ndss-paper/shadow-attacks-hiding-and-replacing-content-in-signed-pdfs/) from the original on April 21, 2021. Retrieved April 6, 2021.
0. "Processing Dangerous Paths – On Security and Privacy of the Portable Document Format" (https://www.ndss-symposium.org/ndss-paper/processing-dangerous-paths-on-security-and-privacy-of-the-portable-document-format/). Internet Society, The Network and Distributed System Security Symposium. Archived (https://web.archive.org/web/20210421094018/https://www.ndss-symposium.org/ndss-paper/processing-dangerous-paths-on-security-and-privacy-of-the-portable-document-format/) from the original on April 21, 2021. Retrieved April 6, 2021.
0. "Portable Document Flaws 101" (https://www.blackhat.com/us-20/briefings/schedule/#portable-document-flaws--20387). Blackhat. Archived (https://web.archive.org/web/20210409131634/https://www.blackhat.com/us-20/briefings/schedule/#portable-document-flaws--20387) from the original on April 9, 2021. Retrieved April 6, 2021.
0. "Can PDFs have viruses? Keep your files safe" (https://www.adobe.com/acrobat/resources/can-pdfs-contain-viruses.html). Adobe. Archived (https://web.archive.org/web/20231004120143/https://www.adobe.com/acrobat/resources/can-pdfs-contain-viruses.html) from the original on October 4, 2023. Retrieved October 3, 2023.
0. Adobe Forums, Announcement: PDF Attachment Virus "Peachy" (https://forums.adobe.com/thread/302989) Archived (https://web.archive.org/web/20150904151955/https://forums.adobe.com/thread/302989) September 4, 2015, at the Wayback Machine, August 15, 2001.
0. "Security bulletins and advisories" (https://helpx.adobe.com/security.html#readerwin). Adobe Systems Inc. January 10, 2023. Archived (https://web.archive.org/web/20100406041941/http://www.adobe.com/support/security/#readerwin) from the original on April 6, 2010. Retrieved January 12, 2023.
0. Gibson, Steve; Laporte, Leo (March 12, 2009). "Steve Gibson – SecurityNow Podcast" (https://www.grc.com/sn/sn-187.txt). Archived (https://web.archive.org/web/20200508100301/https://www.grc.com/sn/sn-187.txt) from the original on May 8, 2020. Retrieved January 11, 2011.
0. "Malicious PDFs Execute Code Without a Vulnerability" (https://web.archive.org/web/20100404034752/http://blogs.pcmag.com/securitywatch/2010/03/malicious_pdfs_execute_code_wi.php). *PCMAG*. Archived from the original (http://blogs.pcmag.com/securitywatch/2010/03/malicious_pdfs_execute_code_wi.php) on April 4, 2010.

0. Pathak, Khamosh (October 7, 2017). "How to Create a PDF from Web Page on iPhone and iPad in iOS 11" (https://ijunkie.com/how-to-create-pdf-web-page-safari-iphone-ipad-ios-11/). *iJunkie*. Archived (https://web.archive.org/web/20230112153246/https://ijunkie.com/how-to-create-pdf-web-page-safari-iphone-ipad-ios-11/) from the original on January 12, 2023. Retrieved January 12, 2023.
0. "Description of 2007 Microsoft Office Suite Service Pack 2 (SP2)" (https://web.archive.org/web/20090429212434/http://support.microsoft.com/kb/953195). Microsoft. Archived from the original (http://support.microsoft.com/kb/953195) on April 29, 2009. Retrieved January 12, 2023.
0. On 2014-04-02, a note dated February 10, 2009 referred to Current FSF High Priority Free Software Projects (http://www.fsf.org/campaigns/priority.html) Archived (https://web.archive.org/web/20070810230457/http://www.fsf.org/campaigns/priority.html) August 10, 2007, at the Wayback Machine as a source. Content of the latter page, however, changes over time.
0. "Goals and Motivations" (https://web.archive.org/web/20140704114405/http://www.gnupdf.org/Goals_and_Motivations). *gnupdf.org*. GNUpdf. November 28, 2007. Archived from the original on July 4, 2014. Retrieved April 2, 2014.
0. Lee, Matt (October 6, 2011). "GNU PDF project leaves FSF High Priority Projects list; mission complete!" (http://www.fsf.org/blogs/community/gnu-pdf-project-leaves-high-priority-projects-list-mission-complete). *fsf.org*. Free Software Foundation. Archived (https://web.archive.org/web/20141228050435/http://www.fsf.org/blogs/community/gnu-pdf-project-leaves-high-priority-projects-list-mission-complete) from the original on December 28, 2014.
0. "Poppler Homepage" (http://poppler.freedesktop.org/). Archived (https://web.archive.org/web/20150108235708/http://poppler.freedesktop.org/) from the original on January 8, 2015. Retrieved January 12, 2023. "Poppler is a PDF rendering library based on the xpdf-3.0 code base."
0. "Xpdf License" (http://cgit.freedesktop.org/poppler/poppler/tree/README-XPDF). Archived (https://archive.today/20130414194348/http://cgit.freedesktop.org/poppler/poppler/tree/README-XPDF) from the original on April 14, 2013. Retrieved January 12, 2023. "Xpdf is licensed under the GNU General Public License (GPL), version 2 or 3."
0. "The Apache PDFBox project- Apache PDFBox 3.0.0 released" (https://pdfbox.apache.org/). August 17, 2023. Archived (https://web.archive.org/web/20230107234923/https://pdfbox.apache.org/) from the original on January 7, 2023. Updated for new releases.
0. "Adobe PDF Print Engine" (https://www.adobe.com/products/pdfprintengine/overview.html). Adobe Systems Inc. Archived (https://web.archive.org/web/20130822034446/http://www.adobe.com/products/pdfprintengine/overview.html) from the original on August 22, 2013. Retrieved August 20, 2014.

0. "Jaws® 3.0 PDF and PostScript RIP SDK" (https://web.archive.org/web/20160305090728/http://globalgraphics.com/products/jaws_rip). *globalgraphics.com*. Archived from the original (http://www.globalgraphics.com/products/jaws_rip/) on March 5, 2016. Retrieved November 26, 2010.
0. "Harlequin MultiRIP" (https://web.archive.org/web/20140209215413/http://www.globalgraphics.com/products/harlequin-multi-rip/). Archived from the original (http://www.globalgraphics.com/products/harlequin-multi-rip) on February 9, 2014. Retrieved March 2, 2014.
0. "Press-Ready PDF Files" (https://web.archive.org/web/20090205151505/http://www.prepressx.com/). Archived from the original on February 5, 2009. Retrieved January 12, 2023. "For anyone interested in having their graphic project commercially printed directly from digital files or PDFs."
0. "PDF as Standard Print Job Format" (https://web.archive.org/web/20091114130224/https://www.linuxfoundation.org/collaborate/workgroups/openprinting/pdf_as_standard_print_job_format). *The Linux Foundation*. Linux Foundation. October 23, 2009. Archived from the original (http://www.linuxfoundation.org/collaborate/workgroups/openprinting/pdf_as_standard_print_job_format) on November 14, 2009. Retrieved January 12, 2023.
0. Meuschke, Norman; Jagdale, Apurva; Spinde, Timo; Mitrović, Jelena; Gipp, Bela (2023), Sserwanga, Isaac; Goulding, Anne; Moulaison-Sandy, Heather; Du, Jia Tina (eds.), "A Benchmark of PDF Information Extraction Tools Using a Multi-task and Multi-domain Evaluation Framework for Academic Documents" (https://link.springer.com/10.1007/978-3-031-28032-0_31), *Information for a Better World: Normality, Virtuality, Physicality, Inclusivity*, vol. 13972, Cham: Springer Nature Switzerland, pp. 383–405, arXiv:2303.09957 (https://arxiv.org/abs/2303.09957), doi:10.1007/978-3-031-28032-0_31 (https://doi.org/10.1007%2F978-3-031-28032-0_31), ISBN 978-3-031-28031-3

Further reading

ISO Standards

- PDF 2.0 "ISO 32000-2:2020(en), Document management — Portable document format — Part 2: PDF 2.0" (https://www.iso.org/standard/75839.html). *International Organization for Standardization*. Retrieved December 16, 2020.
- PDF 2.0 "ISO 32000-2:2017(en), Document management — Portable document format — Part 2: PDF 2.0" (https://www.iso.org/standard/63534.html). *International Organization for Standardization*. August 3, 2017. Retrieved January 31, 2019.

Adobe open source standards

- PDF 1.7 (ISO 32000-1:2008) (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/PDF32000_2008.pdf)
- PDF 1.7 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.7old.pdf) and errata to 1.7 (https://web.archive.org/web/20220306202833/https://www.adobe.com/content/dam/acom/en/devnet/pdf/pdf_reference_archive/pdf_17_errata.pdf) at the Wayback Machine (archived March 6, 2022)
- PDF 1.6 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.6.pdf) (ISBN 0-321-30474-8) and errata to 1.6 (https://web.archive.org/web/20220306152230/https://www.adobe.com/content/dam/acom/en/devnet/pdf/pdf_reference_archive/PDF16Errata.pdf) at the Wayback Machine (archived March 6, 2022)
- PDF 1.5 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.5_v6.pdf) and errata to 1.5 (https://web.archive.org/web/20211222122128/https://www.adobe.com/content/dam/acom/en/devnet/pdf/pdf_reference_archive/errata.txt) at the Wayback Machine (archived December 22, 2021)
- PDF 1.4 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.4.pdf) (ISBN 0-201-75839-3) and errata to 1.4 (https://web.archive.org/web/20220306152229/https://www.adobe.com/content/dam/acom/en/devnet/pdf/pdf_reference_archive/PDF14errata.txt) at the Wayback Machine (archived March 6, 2022)
- PDF 1.3 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.3.pdf) (ISBN 0-201-61588-6) and errata to 1.3 (https://web.archive.org/web/20220306152234/https://www.adobe.com/content/dam/acom/en/devnet/pdf/pdfs/PDFerrata.txt) at the Wayback Machine (archived March 6, 2022)
- PDF 1.2 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.2.pdf)
- PDF 1.0 (https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.0.pdf) (ISBN 0-201-62628-4)

Conference papers

- Hardy, M. R. B.; Brailsford, D. F. (2002). "Mapping and displaying structural transformations between XML and PDF" (https://web.archive.org/web/20170324072906/https://www.cs.nott.ac.uk/~psadb1/Publications/Download/2002/Hardy02.pdf) (PDF). *Proceedings of the 2002 ACM symposium on Document engineering – DocEng '02*. pp. 95–102. doi:10.1145/585058.585077 (https://doi.org/10.1145%2F585058.585077). ISBN 1-58113-594-7. S2CID 9371237 (https://api.semanticscholar.org/CorpusID:9371237). Archived from the original

(https://www.cs.nott.ac.uk/~psadb1/Publications/Download/2002/Hardy02.pdf) (PDF) on March 24, 2017.

External links

- PDF Association (https://pdfa.org/) – The PDF Association is the industry association for software developers producing or processing PDF files.
  - PDF Specification Index (https://pdfa.org/resource/pdf-specification-index/) at the PDF Association
  - PDF Cheat Sheets, 2nd edition (https://pdfa.org/resource/pdf-cheat-sheets/) by the PDF Association (last updated October 29, 2024)
  - Sponsored free access to the ISO 32000-2 (PDF 2.0) bundle (https://www.pdfa-inc.org/product/iso-32000-2-pdf-2-0-bundle-sponsored-access/), including the latest core PDF specification and five ISO standardized extensions to the core specification
- Format description of the PDF family (https://www.loc.gov/preservation/digital/formats/fdd/fdd000030.shtml), PDF/A (https://www.loc.gov/preservation/digital/formats/fdd/fdd000318.shtml), PDF/X (https://www.loc.gov/preservation/digital/formats/fdd/fdd000124.shtml) from Library of Congress

Tech notes from Adobe

- Adobe PDF 101: Summary of PDF (https://web.archive.org/web/20101007220449/http://partners.adobe.com/public/developer/tips/topic_tip31.html) at the Wayback Machine (archived 2010-10-07)
- Adobe: PostScript vs. PDF (https://web.archive.org/web/20160413212438/https://www.adobe.com/print/features/psvspdf/) at the Wayback Machine (archived 2016-04-13) – Official introductory comparison of PS, EPS vs. PDF.
- PDF Reference and Adobe Extensions to the PDF Specification (https://web.archive.org/web/20210116133007/https://www.adobe.com/devnet/pdf/pdf_reference.html) at the Wayback Machine (archived 2021-01-16)

Articles

- Portable Document Format: An Introduction for Programmers (1999) (http://preserve.mactech.com/articles/mactech/Vol.15/15.09/PDFIntro/index.html) from MacTech – Introduction to PDF vs. PostScript and PDF internals (up to v1.3)
- John Warnock's 'Camelot' signalled birth of PDF (2002) (https://web.archive.org/web/20190422013101/http://www.planetpdf.com/enterprise/article.asp?ContentID=6519) at the Wayback Machine (archived 2019-04-22) from Planet PDF that describes the paper in which John Warnock outlined the project that created PDF

Videos

- Video: Everything you wanted to know about PDF but was afraid to ask (https://www.youtube.com/watch?v=poc9PVmFzpc) — Recording of a talk by Leonard Rosenthol (Adobe Systems) at TUG 2007

HTML

HTML

Official logo of HTML5[1]

Filename extension	`.html` `.htm`
Internet media type	`text/html`
Type code	TEXT
Uniform Type Identifier (UTI)	public.html
Developed by	WHATWG World Wide Web Consortium (W3C; formerly)
Initial release	1993
Latest release	Living Standard (https://html.spec.whatwg.org/multipage/)
Type of format	Document file format
Container for	HTML elements
Contained by	Web browser
Extended from	SGML
Extended to	XHTML
Open format?	Yes
Website	html.spec.whatwg.org (https://html.spec.whatwg.org/)

Hypertext Markup Language (**HTML**) is the standard markup language[a] for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript, a programming language.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes, and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as `<img>` and `<input>` directly introduce content into

the page. Other tags such as `<p>` and `</p>` surround and provide information about document text and may include sub-element tags. Browsers do not display the HTML tags but use them to interpret the content of the page.
HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. The inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.[3] A form of HTML, known as HTML5, is used to display video and audio, primarily using the `<canvas>` element, together with JavaScript.

History

Development


Tim Berners-Lee in April 2009

In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system.[4] Berners-Lee specified HTML and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes of 1990, Berners-Lee listed "some of the many areas in which hypertext is used"; an encyclopedia is the first entry.[5]
The first publicly available description of HTML was a document called "HTML Tags",[6] first mentioned on the Internet by Tim Berners-Lee in late 1991.[7][8] It describes 18 elements comprising the initial, relatively simple design of HTML. Except for the hyperlink tag, these were strongly influenced by CERN SGML, an in-house Standard Generalized Markup Language (SGML)-based documentation format at CERN. Eleven of these elements still exist in HTML 4.[9]
HTML is a markup language that web browsers use to interpret and compose text, images, and other material into visible or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS. Many of the text elements are mentioned in the 1988 ISO technical report TR 9537 *Techniques for using SGML*, which describes the features of early text formatting languages such as that used by the RUNOFF command developed in the early 1960s for the CTSS (Compatible Time-Sharing System) operating system. These formatting commands were derived from the commands used by typesetters to manually format documents.

However, the SGML concept of generalized markup is based on elements (nested annotated ranges with attributes) rather than merely print effects, with separate structure and markup. HTML has been progressively moved in this direction with CSS.
Berners-Lee considered HTML to be an application of SGML. It was formally defined as such by the Internet Engineering Task Force (IETF) with the mid-1993 publication of the first proposal for an HTML specification, the "Hypertext Markup Language (HTML)" Internet Draft by Berners-Lee and Dan Connolly, which included an SGML Document type definition to define the syntax.[10][11] The draft expired after six months, but was notable for its acknowledgment of the NCSA Mosaic browser's custom tag for embedding in-line images, reflecting the IETF's philosophy of basing standards on successful prototypes. Similarly, Dave Raggett's competing Internet Draft, "HTML+ (Hypertext Markup Format)", from late 1993, suggested standardizing already-implemented features like tables and fill-out forms.[12]
After the HTML and HTML+ drafts expired in early 1994, the IETF created an HTML Working Group. In 1995, this working group completed "HTML 2.0", the first HTML specification intended to be treated as a standard against which future implementations should be based.[13]
Further development under the auspices of the IETF was stalled by competing interests. Since 1996, the HTML specifications have been maintained, with input from commercial software vendors, by the World Wide Web Consortium (W3C).[14] In 2000, HTML became an international standard (ISO/IEC 15445:2000). HTML 4.01 was published in late 1999, with further errata published through 2001. In 2004, development began on HTML5 in the Web Hypertext Application Technology Working Group (WHATWG), which became a joint deliverable with the W3C in 2008, and was completed and standardized on 28 October 2014.[15]

HTML version timeline

HTML 2

November 24, 1995

HTML 2.0 was published as RFC 1866 (https://datatracker.ietf.org/doc/html/rfc1866). Supplemental RFCs added capabilities:

- November 25, 1995: RFC 1867 (https://datatracker.ietf.org/doc/html/rfc1867) (form-based file upload)
- May 1996: RFC 1942 (https://datatracker.ietf.org/doc/html/rfc1942) (tables)
- August 1996: RFC 1980 (https://datatracker.ietf.org/doc/html/rfc1980) (client-side image maps)
- January 1997: RFC 2070 (https://datatracker.ietf.org/doc/html/rfc2070) (internationalization)

HTML 3

January 14, 1997

HTML 3.2[16] was published as a W3C Recommendation. It was the first version developed and standardized exclusively by the W3C, as the IETF had closed its HTML Working Group on September 12, 1996.[17]
Initially code-named "Wilbur",[18] HTML 3.2 dropped math formulas entirely, reconciled overlap among various proprietary extensions and adopted most of Netscape's visual markup tags. Netscape's blink element and Microsoft's marquee element were omitted due to a mutual agreement between the two companies.[14] A markup for mathematical formulas similar to that of HTML was standardized 14 months later in MathML.

HTML 4

December 18, 1997
HTML 4.0[19] was published as a W3C Recommendation. It offers three variations:

- **Strict, in which deprecated elements are forbidden**
- **Transitional, in which deprecated elements are allowed**
- **Frameset, in which mostly only frame related elements are allowed.**

Initially code-named "Cougar",[18] HTML 4.0 adopted many browser-specific element types and attributes, but also sought to phase out Netscape's visual markup features by marking them as deprecated in favor of style sheets. HTML 4 is an SGML application conforming to ISO 8879 – SGML.[20]
April 24, 1998
HTML 4.0[21] was reissued with minor edits without incrementing the version number.
December 24, 1999
HTML 4.01[22] was published as a W3C Recommendation. It offers the same three variations as HTML 4.0 and its last errata[23] were published on May 12, 2001.
May 2000
ISO/IEC 15445:2000[24] ("ISO HTML", based on HTML 4.01 Strict) was published as an ISO/IEC international standard.[25] In the ISO, this standard is in the domain of the ISO/IEC JTC 1/SC 34 (ISO/IEC Joint Technical Committee 1, Subcommittee 34 – Document description and processing languages).[24]
After HTML 4.01, there were no new versions of HTML for many years, as the development of the parallel, XML-based language XHTML occupied the W3C's HTML Working Group.

HTML 5

October 28, 2014

HTML5[26] was published as a W3C Recommendation.[27]

November 1, 2016

HTML 5.1[28] was published as a W3C Recommendation.[29][30]

December 14, 2017

HTML 5.2[31] was published as a W3C Recommendation.[32][33]

HTML draft version timeline

October 1991

HTML Tags,[7] an informal CERN document listing 18 HTML tags, was first mentioned in public.

June 1992

First informal draft of the HTML DTD,[34] with seven subsequent revisions (July 15, August 6, August 18, November 17, November 19, November 20, November 22)[35][36][37]

November 1992

HTML DTD 1.1 (the first with a version number, based on RCS revisions, which start with 1.1 rather than 1.0), an informal draft[37]

June 1993

Hypertext Markup Language[38] was published by the IETF IIIR Working Group as an Internet Draft (a rough proposal for a standard). It was replaced by a second version[39] one month later.

November 1993

HTML+ (https://www.w3.org/MarkUp/HTMLPlus/htmlplus_1.html) was published by the IETF as an Internet Draft and was a competing proposal to the Hypertext Markup Language draft. It expired in July 1994.[40]

November 1994

First draft (revision 00) of HTML 2.0 published by IETF itself[41] (called as "HTML 2.0" from revision 02[42]), that finally led to the publication of RFC 1866 (https://datatracker.ietf.org/doc/html/rfc1866) in November 1995.[43]

April 1995 (authored March 1995)

HTML 3.0[44] was proposed as a standard to the IETF, but the proposal expired five months later (28 September 1995)[45] without further action. It included many of the capabilities that were in Raggett's HTML+ proposal, such as support for tables, text flow around figures, and the display of complex mathematical formulas.[45]

W3C began development of its own Arena browser as a test bed for HTML 3 and Cascading Style Sheets,[46][47][48] but HTML 3.0 did not succeed for several reasons. The draft was considered very large at 150 pages and the pace of browser development, as well as the number of interested parties, had outstripped the resources of the IETF.[14] Browser vendors, including Microsoft and Netscape at the time, chose to implement different subsets of HTML 3's draft features as well as to introduce their own extensions to it.[14]

(See browser wars.) These included extensions to control stylistic aspects of documents, contrary to the "belief [of the academic engineering community] that such things as text color, background texture, font size, and font face were definitely outside the scope of a language when their only intent was to specify how a document would be organized."[14] Dave Raggett, who has been a W3C Fellow for many years, has commented for example: "To a certain extent, Microsoft built its business on the Web by extending HTML features."[14]

January 2008

HTML5 was published as a Working Draft by the W3C.[49]

Logo of HTML5

Although its syntax closely resembles that of SGML, HTML5 has abandoned any attempt to be an SGML application and has explicitly defined its own "html" serialization, in addition to an alternative XML-based XHTML5 serialization.[50]

2011 HTML5 – Last Call

On 14 February 2011, the W3C extended the charter of its HTML Working Group with clear milestones for HTML5. In May 2011, the working group advanced HTML5 to "Last Call", an invitation to communities inside and outside W3C to confirm the technical soundness of the specification. The W3C developed a comprehensive test suite to achieve broad interoperability for the full specification by 2014, which was the target date for recommendation.[51] In January 2011, the WHATWG renamed its "HTML5" living standard to "HTML". The W3C nevertheless continued its project to release HTML5.[52]

2012 HTML5 – Candidate Recommendation

In July 2012, WHATWG and W3C decided on a degree of separation. W3C will continue the HTML5 specification work, focusing on a single definitive standard, which is considered a "snapshot" by WHATWG. The WHATWG organization will continue its work with HTML5 as a "Living Standard". The concept of a living standard is that it is never complete and is always being updated and improved. New features can be added but functionality will not be removed.[53]

In December 2012, W3C designated HTML5 as a Candidate Recommendation.[54] The criterion for advancement to W3C Recommendation is "two 100% complete and fully interoperable implementations".[55]

2014 HTML5 – Proposed Recommendation and Recommendation

In September 2014, W3C moved HTML5 to Proposed Recommendation.[56]
On 28 October 2014, HTML5 was released as a stable W3C Recommendation,[57] meaning the specification process is complete.[58]

XHTML versions

XHTML is a separate language that began as a reformulation of HTML 4.01 using XML 1.0. It is now referred to as *the XML syntax for HTML* and is no longer being developed as a separate standard.[59]

- XHTML 1.0 was published as a W3C Recommendation on January 26, 2000,[60] and was later revised and republished on August 1, 2002. It offers the same three variations as HTML 4.0 and 4.01, reformulated in XML, with minor restrictions.
- XHTML 1.1[61] was published as a W3C Recommendation on May 31, 2001. It is based on XHTML 1.0 Strict, but includes minor changes, can be customized, and is reformulated using modules in the W3C recommendation "Modularization of XHTML", which was published on April 10, 2001.[62]
- XHTML 2.0 was a working draft. Work on it was abandoned in 2009 in favor of work on HTML5 and XHTML5.[63][64][65] XHTML 2.0 was incompatible with XHTML 1.x and, therefore, would be more accurately characterized as an XHTML-inspired new language than an update to XHTML 1.x.

Transition of HTML publication to WHATWG

On 28 May 2019, the W3C announced that WHATWG would be the sole publisher of the HTML and DOM standards.[66][67][68][69] The W3C and WHATWG had been publishing competing standards since 2012. While the W3C standard was identical to the WHATWG in 2007 the standards have since progressively diverged due to different design decisions.[70] The WHATWG "Living Standard" had been the *de facto* web standard for some time.[71]

Markup

HTML markup consists of several key components, including those called *tags* (and their *attributes*), character-based *data types*, *character references* and *entity references*. HTML tags most commonly come in pairs like `<h1>` and `</h1>`, although some represent *empty elements* and so are unpaired, for example `<img>`. The first tag in such a pair is the *start tag*, and the second is the *end tag* (they are also called *opening tags* and *closing tags*).

Another important component is the HTML *document type declaration*, which triggers standards mode rendering.

The following is an example of the classic "Hello, World!" program:

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <div>
        <p>Hello world!</p>
    </div>
```

```
  </body>
</html>
```

The text between `<html>` and `</html>` describes the web page, and the text between `<body>` and `</body>` is the visible page content. The markup text `<title>This is a title</title>` defines the browser page title shown on browser tabs and window titles and the tag `<div>` defines a division of the page used for easy styling. Between `<head>` and `</head>`, a `<meta>` element can be used to define webpage metadata.

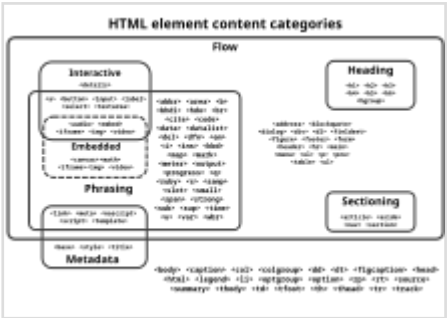
The Document Type Declaration `<!DOCTYPE html>` is for HTML5. If a declaration is not included, various browsers will revert to "quirks mode" for rendering.[72]

Elements

HTML documents imply a structure of nested HTML elements. These are indicated in the document by HTML *tags*, enclosed in angle brackets.[73]

In the simple, general case, the extent of an element is indicated by a pair of tags: a "start tag" `<p>` and "end tag" `</p>`. The text content of the element, if any, is placed between these tags.

Tags may also enclose further tag markup between the start and end, including a mixture of tags and text. This indicates further (nested) elements, as children of the parent element.



HTML element content categories

The start tag may also include the element's *attributes* within the tag. These indicate other information, such as identifiers for sections within the document, identifiers used to bind style information to the presentation of the document, and for some tags such as the `<img>` used to embed images, the reference to the image resource in the format like this: `<img src="example.com/example.jpg">`

Some elements, such as the line break `<br>` do not permit *any* embedded content, either text or further tags. These require only a single empty tag (akin to a start tag) and do not use an end tag.

Many tags, particularly the closing end tag for the very commonly used paragraph element `<p>`, are optional. An HTML browser or other agent can infer the closure for the end of an element from the context and the structural rules defined by the HTML standard. These rules are complex and not widely understood by most HTML authors.

The general form of an HTML element is therefore: `<tag attribute1="value1" attribute2="value2">''content''</tag>`. Some HTML elements are defined as *empty elements* and take the form `<tag attribute1="value1" attribute2="value2">`. Empty elements may enclose no content, for instance, the `<br>` tag or the inline `<img>` tag. The name of an HTML element is the name used in the tags. The end tag's name is preceded by a slash character `/`. If a tag has no content, an end tag is not allowed. If attributes are not mentioned, default values are used in each case.

Element examples

Header of the HTML document: `<head>...</head>`. The title is included in the head, for example:

```
<head>
  <title>The Title</title>
  <link rel="stylesheet" href="stylebyjimbowales.css"> <!-- Imports Stylesheets -->
</head>
```

Headings

HTML headings are defined with the `<h1>` to `<h6>` tags with H1 being the highest (or most important) level and H6 the least:

```
<h1>Heading level 1</h1>
<h2>Heading level 2</h2>
<h3>Heading level 3</h3>
<h4>Heading level 4</h4>
<h5>Heading level 5</h5>
<h6>Heading level 6</h6>
```

The effects are:

Heading Level 1

Heading Level 2

Heading Level 3

Heading Level 4

Heading Level 5

Heading Level 6

CSS can substantially change the rendering.
Paragraphs:

```
<p>Paragraph 1</p> <p>Paragraph 2</p>
```

Line breaks

`<br>`. The difference between `<br>` and `<p>` is that `<br>` breaks a line without altering the semantic structure of the page, whereas `<p>` sections the page into paragraphs. The element `<br>` is an *empty element* in that, although it may have attributes, it can take no content and it must not have an end tag.

```
<p>This <br> is a paragraph <br> with <br> line breaks</p>
```

Links

This is a link in HTML. To create a link the `<a>` tag is used. The `href` attribute holds the URL address of the link.

```
<a href="https://www.wikipedia.org/">A link to Wikipedia!</a>
```

Inputs

There are many possible ways a user can give inputs like:

```
<input type="text"> <!-- This is for text input -->
<input type="file"> <!-- This is for uploading files -->
<input type="checkbox"> <!-- This is for checkboxes -->
```

Comments:

```
<!-- This is a comment -->
```

Comments can help in the understanding of the markup and do not display in the webpage.

There are several types of markup elements used in HTML:

Structural markup indicates the purpose of text

For example, `<h2>Golf</h2>` establishes "Golf" as a second-level heading. Structural markup does not denote any specific rendering, but most web browsers have default styles for element formatting. Content may be further styled using Cascading Style Sheets (CSS).[74]

Presentational markup indicates the appearance of the text, regardless of its purpose

For example, `<b>bold text</b>` indicates that visual output devices should render "boldface" in bold text, but gives a little indication what devices that are unable to do this (such as aural devices that read the text aloud) should do. In the case of both `<b>bold text</b>` and `<i>italic text</i>`, there are other elements that may have equivalent visual renderings but that are more semantic in nature, such as `<strong>strong text</strong>` and `<em>emphasized text</em>` respectively. It is easier to see how an aural user agent should interpret the latter two elements. However, they are not equivalent to their presentational counterparts: it would be undesirable for a screen reader to emphasize the name of a book, for instance, but on a screen, such a name would be italicized. Most presentational markup elements have become deprecated under the HTML 4.0 specification in favor of using CSS for styling.

Hypertext markup makes parts of a document into links to other documents

An anchor element creates a hyperlink in the document and its `href` attribute sets the link's target URL. For example, the HTML markup `<a href="https://en.wikipedia.org/">Wikipedia</a>`, will render the word "Wikipedia (https://en.wikipedia.org/)" as a hyperlink. To render an image as a hyperlink, an `img` element is inserted as content into the a element. Like `br`, `img` is an empty element with attributes but no content or closing tag. `<a href="https://example.org"><img src="image.gif" alt="descriptive text" width="50" height="50" border="0"></a>`.

Attributes

Most of the attributes of an element are name–value pairs, separated by `=` and written within the start tag of an element after the element's name. The value may be enclosed in single or double quotes, although values consisting of certain characters can be left unquoted in HTML (but not XHTML).[75][76] Leaving attribute values unquoted is considered unsafe.[77] In contrast with name-value pair attributes, there are some attributes that affect the element simply by their presence in the start tag of the element,[7] like the `ismap` attribute for the `img` element.[78] There are several common attributes that may appear in many elements :

- The `id` attribute provides a document-wide unique identifier for an element. This is used to identify the element so that stylesheets can alter its presentational properties, and scripts may alter, animate or delete its contents or presentation. Appended to the URL of the page, it provides a globally unique identifier for the element, typically a sub-section of the page. For example, the ID "Attributes" in `https://en.wikipedia.org/wiki/HTML#Attributes`.
- The `class` attribute provides a way of classifying similar elements. This can be used for semantic or presentation purposes. For example, an HTML document might semantically use the designation `<class="notation">` to indicate that all elements with this class value are subordinate to the main text of the document. In presentation, such elements might be gathered together and presented as footnotes on a page instead of appearing in the place where they occur in the HTML source. Class attributes are used semantically in microformats. Multiple class values may be specified; for example `<class="notation important">` puts the element into both the `notation` and the `important` classes.
- An author may use the `style` attribute to assign presentational properties to a particular element. It is considered better practice to use an element's `id` or `class` attributes to select the element from within a stylesheet, though sometimes this can be too cumbersome for a simple, specific, or ad hoc styling.
- The `title` attribute is used to attach a subtextual explanation to an element. In most browsers this attribute is displayed as a tooltip.

- The `lang` attribute identifies the natural language of the element's contents, which may be different from that of the rest of the document. For example, in an English-language document:

```
<p>Oh well, <span lang="fr">c'est la vie</span>, as they say in France.</p>
```

The abbreviation element, `abbr`, can be used to demonstrate some of these attributes:

```
<abbr id="anId" class="jargon" style="color:purple;" title="Hypertext Markup Language">HTML</
abbr>
```

This example displays as HTML; in most browsers, pointing the cursor at the abbreviation should display the title text "Hypertext Markup Language."

Most elements take the language-related attribute `dir` to specify text direction, such as with "rtl" for right-to-left text in, for example, Arabic, Persian or Hebrew.[79]

Character and entity references

As of version 4.0, HTML defines a set of 252 character entity references and a set of 1,114,050 numeric character references, both of which allow individual characters to be written via simple markup, rather than literally. A literal character and its markup counterpart are considered equivalent and are rendered identically.

The ability to "escape" characters in this way allows for the characters `<` and `&` (when written as `<` and `&`, respectively) to be interpreted as character data, rather than markup. For example, a literal `<` normally indicates the start of a tag, and `&` normally indicates the start of a character entity reference or numeric character reference; writing it as `&` or `&#x26;` or `&#38;` allows `&` to be included in the content of an element or in the value of an attribute. The double-quote character (`"`), when not used to quote an attribute value, must also be escaped as `&quot;` or `&#x22;` or `&#34;` when it appears within the attribute value itself. Equivalently, the single-quote character (`'`), when not used to quote an attribute value, must also be escaped as `&#x27;` or `&#39;` (or as `&apos;` in HTML5 or XHTML documents[80][81]) when it appears within the attribute value itself. If document authors overlook the need to escape such characters, some browsers can be very forgiving and try to use context to guess their intent. The result is still invalid markup, which makes the document less accessible to other browsers and to other user agents that may try to parse the document for search and indexing purposes for example.

Escaping also allows for characters that are not easily typed, or that are not available in the document's character encoding, to be represented within the element and attribute content. For example, the acute-accented `e` (é), a character typically found only on Western European and South American keyboards, can be written in any HTML document as the entity reference `&eacute;` or as the numeric references `&#xE9;` or `&#233;`, using characters that are available on all keyboards and are supported in all character encodings. Unicode character encodings such as UTF-8 are compatible with all modern browsers and allow direct access to almost all the characters of the world's writing systems.[82]

Example HTML Escape Sequences

Named	Decimal	Hexadecimal	Result	Description	Notes
&amp;	&#38;	&#x26;	&	Ampersand	
&lt;	&#60;	&#x3C;	<	Less Than	
&gt;	&#62;	&#x3E;	>	Greater Than	
&quot;	&#34;	&#x22;	"	Double Quote	
&apos;	&#39;	&#x27;	'	Single Quote	
				Non-Breaking Space	
&copy;	&#169;	&#xA9;	©	Copyright	
&reg;	&#174;	&#xAE;	®	Registered Trademark	
&dagger;	&#8224;	&#x2020;	†	Dagger	
&ddagger;	&#8225;	&#x2021;	‡	Double dagger	Names are case-sensitive and may have synonyms.
&trade;	&#8482;	&#x2122;	™	Trademark	

Data types

HTML defines several data types for element content, such as script data and stylesheet data, and a plethora of types for attribute values, including IDs, names, URIs, numbers, units of length, languages, media descriptors, colors, character encodings, dates and times, and so on. All of these data types are specializations of character data.

Document type declaration

HTML documents are required to start with a document type declaration (informally, a "doctype"). In browsers, the doctype helps to define the rendering mode—particularly whether to use quirks mode.

The original purpose of the doctype was to enable the parsing and validation of HTML documents by SGML tools based on the document type definition (DTD). The DTD to which the DOCTYPE refers contains a machine-readable grammar specifying the permitted and prohibited content for a document conforming to such a DTD. Browsers, on the other hand, do not implement HTML as an application of SGML and as consequence do not read the DTD.

HTML5 does not define a DTD; therefore, in HTML5 the doctype declaration is simpler and shorter:[83]

```
<!DOCTYPE html>
```

An example of an HTML 4 doctype

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "https://www.w3.org/TR/html4/strict.dtd">
```

This declaration references the DTD for the "strict" version of HTML 4.01. SGML-based validators read the DTD in order to properly parse the document and to perform validation. In modern browsers, a valid doctype activates standards mode as opposed to quirks mode.

In addition, HTML 4.01 provides Transitional and Frameset DTDs, as explained below. The transitional type is the most inclusive, incorporating current tags as well as older or "deprecated" tags, with the Strict DTD excluding deprecated tags. The frameset has all tags necessary to make frames on a page along with the tags included in transitional type.[84]

Semantic HTML

Semantic HTML is a way of writing HTML that emphasizes the meaning of the encoded information over its presentation (look). HTML has included semantic markup from its inception,[85] but has also included presentational markup, such as `<font>`, `<i>` and `<center>` tags. There are also the semantically neutral div and span tags. Since the late 1990s, when Cascading Style Sheets were beginning to work in most browsers, web authors have been encouraged to avoid the use of presentational HTML markup with a view to the separation of content and presentation.[86]

In a 2001 discussion of the Semantic Web, Tim Berners-Lee and others gave examples of ways in which intelligent software "agents" may one day automatically crawl the web and find, filter, and correlate previously unrelated, published facts for the benefit of human users.[87] Such agents are not commonplace even now, but some of the ideas of Web 2.0, mashups and price comparison websites may be coming close. The main difference between these web application hybrids and Berners-Lee's semantic agents lies in the fact that the current aggregation and hybridization of information is usually designed by web developers, who already know the web locations and the API semantics of the specific data they wish to mash, compare and combine.

An important type of web agent that does crawl and read web pages automatically, without prior knowledge of what it might find, is the web crawler or search-engine spider. These software agents are dependent on the semantic clarity of web pages they find as they use various techniques and algorithms to read and index millions of web pages a day and provide web users with search facilities without which the World Wide Web's usefulness would be greatly reduced.

In order for search engine spiders to be able to rate the significance of pieces of text they find in HTML documents, and also for those creating mashups and other hybrids as well as for more automated agents as they are developed, the semantic structures that exist in HTML need to be widely and uniformly applied to bring out the meaning of the published text.[88]

Presentational markup tags are deprecated in current HTML and XHTML recommendations. The majority of presentational features from previous versions of HTML are no longer allowed as they lead to poorer accessibility, higher cost of site maintenance, and larger document sizes.[89]

Good semantic HTML also improves the accessibility of web documents (see also Web Content Accessibility Guidelines). For example, when a screen reader or audio browser can correctly ascertain the structure of a document, it will not waste the visually impaired user's time by reading out repeated or irrelevant information when it has been marked up correctly.

Delivery

HTML documents can be delivered by the same means as any other computer file. However, they are most often delivered either by HTTP from a web server or by email.

HTTP

The World Wide Web is composed primarily of HTML documents transmitted from web servers to web browsers using the Hypertext Transfer Protocol (HTTP). However, HTTP is used to serve images, sound, and other content, in addition to HTML. To allow the web browser to know how to handle each document it receives, other information is transmitted along with the document. This meta data usually includes the MIME type (e.g., `text/html` or `application/xhtml+xml`) and the character encoding (see Character encodings in HTML).

In modern browsers, the MIME type that is sent with the HTML document may affect how the document is initially interpreted. A document sent with the XHTML MIME type is expected to be well-formed XML; syntax errors may cause the browser to fail to render it. The same document sent with the HTML MIME type might be displayed successfully since some browsers are more lenient with HTML.

The W3C recommendations state that XHTML 1.0 documents that follow guidelines set forth in the recommendation's Appendix C may be labeled with either MIME Type.[90] XHTML 1.1 also states that XHTML 1.1 documents should[91] be labeled with either MIME type.[92]

HTML e-mail

Most graphical email clients allow the use of a subset of HTML (often ill-defined) to provide formatting and semantic markup not available with plain text. This may include typographic information like colored headings, emphasized and quoted text, inline images and diagrams. Many such clients include both a GUI editor for composing HTML e-mail messages and a rendering engine for displaying them. Use of HTML in e-mail is criticized by some because of compatibility issues, because it can help disguise phishing attacks, because of accessibility issues for blind or visually impaired people, because it can confuse spam filters and because the message size is larger than plain text.

Naming conventions

The most common filename extension for files containing HTML is `.html`. A common abbreviation of this is `.htm`, which originated because some early operating systems and file systems, such as DOS and the limitations imposed by FAT data structure, limited file extensions to three letters.[93]

HTML Application

An HTML Application (HTA; file extension `.hta`) is a Microsoft Windows application that uses HTML and Dynamic HTML in a browser to provide the application's graphical interface. A regular HTML file is confined to the security model of the web browser's security,

communicating only to web servers and manipulating only web page objects and site cookies. An HTA runs as a fully trusted application and therefore has more privileges, like creation/editing/removal of files and Windows Registry entries. Because they operate outside the browser's security model, HTAs cannot be executed via HTTP, but must be downloaded (just like an EXE file) and executed from local file system.

HTML4 variations

Since its inception, HTML and its associated protocols gained acceptance relatively quickly. However, no clear standards existed in the early years of the language. Though its creators originally conceived of HTML as a semantic language devoid of presentation details,[94] practical uses pushed many presentational elements and attributes into the language, driven largely by the various browser vendors. The latest standards surrounding HTML reflect efforts to overcome the sometimes chaotic development of the language[95] and to create a rational foundation for building both meaningful and well-presented documents. To return HTML to its role as a semantic language, the W3C has developed style languages such as CSS and XSL to shoulder the burden of presentation. In conjunction, the HTML specification has slowly reined in the presentational elements.

There are two axes differentiating various variations of HTML as currently specified: SGML-based HTML versus XML-based HTML (referred to as XHTML) on one axis, and strict versus transitional (loose) versus frameset on the other axis.

SGML-based versus XML-based HTML

One difference in the latest HTML specifications lies in the distinction between the SGML-based specification and the XML-based specification. The XML-based specification is usually called XHTML to distinguish it clearly from the more traditional definition. However, the root element name continues to be "html" even in the XHTML-specified HTML. The W3C intended XHTML 1.0 to be identical to HTML 4.01 except where limitations of XML over the more complex SGML require workarounds. Because XHTML and HTML are closely related, they are sometimes documented in parallel. In such circumstances, some authors conflate the two names as (X)HTML or X(HTML).

Like HTML 4.01, XHTML 1.0 has three sub-specifications: strict, transitional, and frameset.

Aside from the different opening declarations for a document, the differences between an HTML 4.01 and XHTML 1.0 document—in each of the corresponding DTDs—are largely syntactic. The underlying syntax of HTML allows many shortcuts that XHTML does not, such as elements with optional opening or closing tags, and even empty elements which must not have an end tag. By contrast, XHTML requires all elements to have an opening tag and a closing tag. XHTML, however, also introduces a new shortcut: an XHTML tag may be opened and closed within the same tag, by including a slash before the end of the tag like this: `<br/>`. The introduction of this shorthand, which is not used in the SGML declaration for HTML 4.01, may confuse earlier software unfamiliar with this new convention. A fix for this is remove the slash preceding the closing angle bracket, as such: `<br>`.[96]

To understand the subtle differences between HTML and XHTML, consider the transformation of a valid and well-formed XHTML 1.0 document that adheres to Appendix C (see below) into a valid HTML 4.01 document. Making this translation requires the following steps:

1. **The language for an element should be specified with a `lang` attribute rather than the XHTML `xml:lang` attribute.** XHTML uses XML's built-in language-defining functionality attribute.
2. **Remove the XML namespace (`xmlns=URI`).** HTML has no facilities for namespaces.
3. **Change the document type declaration** from XHTML 1.0 to HTML 4.01. (see DTD section for further explanation).
4. If present, **remove the XML declaration.** (Typically this is: `<?xml version="1.0" encoding="utf-8"?>`).
5. **Ensure that the document's MIME type is set to `text/html`.** For both HTML and XHTML, this comes from the HTTP `Content-Type` header sent by the server.
6. **Change the XML empty-element syntax to an HTML style empty element** (`<br/>` to `<br>`).

Those are the main changes necessary to translate a document from XHTML 1.0 to HTML 4.01. To translate from HTML to XHTML would also require the addition of any omitted opening or closing tags. Whether coding in HTML or XHTML it may just be best to always include the optional tags within an HTML document rather than remembering which tags can be omitted.

A well-formed XHTML document adheres to all the syntax requirements of XML. A valid document adheres to the content specification for XHTML, which describes the document structure.

The W3C recommends several conventions to ensure an easy migration between HTML and XHTML (see HTML Compatibility Guidelines (https://www.w3.org/TR/xhtml1/#guidelines)). The following steps can be applied to XHTML 1.0 documents only:

- Include both `xml:lang` and `lang` attributes on any elements assigning language.
- Use the empty-element syntax only for elements specified as empty in HTML.
- Remove the closing slash in empty-element tags: for example `<br>` instead of `<br/>`.
- Include explicit close tags for elements that permit content but are left empty (for example, , not `<div />`).
- Omit the XML declaration.

By carefully following the W3C's compatibility guidelines, a user agent should be able to interpret the document equally as HTML or XHTML. For documents that are XHTML 1.0 and have been made compatible in this way, the W3C permits them to be served either as HTML (with a `text/html` MIME type), or as XHTML (with an `application/xhtml+xml` or `application/xml` MIME type). When delivered as XHTML, browsers should use an XML parser, which adheres strictly to the XML specifications for parsing the document's contents.

Transitional versus strict

HTML 4 defined three different versions of the language: Strict, Transitional (once called Loose), and Frameset. The Strict version is intended for new documents and is considered best practice, while the Transitional and Frameset versions were developed to make it easier to transition documents that conformed to older HTML specifications or did not conform to any specification to a version of HTML 4. The Transitional and Frameset versions allow for presentational markup, which is omitted in the Strict version. Instead, cascading style sheets are encouraged to improve the presentation of HTML documents. Because XHTML 1 only defines an XML syntax for the language defined by HTML 4, the same differences apply to XHTML 1 as well.
The Transitional version allows the following parts of the vocabulary, which are not included in the Strict version:

- **A looser content model**
  - Inline elements and plain text are allowed directly in: body, `blockquote`, `form`, `noscript` and `noframes`
- **Presentation related elements**
  - underline (u) (Deprecated. can confuse a visitor with a hyperlink.)
  - strike-through (`s`)
  - `center` (Deprecated. use CSS instead.)
  - `font` (Deprecated. use CSS instead.)
  - `basefont` (Deprecated. use CSS instead.)
- **Presentation related attributes**
  - `background` (Deprecated. use CSS instead.) and `bgcolor` (Deprecated. use CSS instead.) attributes for body (required element according to the W3C.) element.
  - `align` (Deprecated. use CSS instead.) attribute on `div`, `form`, paragraph (p) and heading (h1...h6) elements
  - `align` (Deprecated. use CSS instead.), `noshade` (Deprecated. use CSS instead.), `size` (Deprecated. use CSS instead.) and `width` (Deprecated. use CSS instead.) attributes on `hr` element
  - `align` (Deprecated. use CSS instead.), `border`, `vspace` and `hspace` attributes on `img` and `object` (caution: the `object` element is only supported in Internet Explorer (from the major browsers)) elements
  - `align` (Deprecated. use CSS instead.) attribute on `legend` and `caption` elements
  - `align` (Deprecated. use CSS instead.) and `bgcolor` (Deprecated. use CSS instead.) on `table` element
  - `nowrap` (Obsolete), `bgcolor` (Deprecated. use CSS instead.), `width`, `height` on `td` and `th` elements
  - `bgcolor` (Deprecated. use CSS instead.) attribute on `tr` element

    - `clear` (Obsolete) attribute on `br` element
    - `compact` attribute on `dl`, `dir` and `menu` elements
    - `type` (Deprecated. use CSS instead.), `compact` (Deprecated. use CSS instead.) and `start` (Deprecated. use CSS instead.) attributes on `ol` and `ul` elements
    - `type` and `value` attributes on `li` element
    - `width` attribute on `pre` element
- **Additional elements in Transitional specification**
    - `menu` (Deprecated. use CSS instead.) list (no substitute, though the unordered list, is recommended)
    - `dir` (Deprecated. use CSS instead.) list (no substitute, though the unordered list is recommended)
    - `isindex` (Deprecated.) (element requires server-side support and is typically added to documents server-side, `form` and `input` elements can be used as a substitute)
    - `applet` (Deprecated. use the `object` element instead.)
- **The `language` (Obsolete) attribute on script element** (redundant with the `type` attribute).
- **Frame related entities**
    - `iframe`
    - `noframes`
    - `target` (Deprecated in the `map`, `link` and `form` elements.) attribute on `a`, client-side image-map (`map`), `link`, `form` and `base` elements

The Frameset version includes everything in the Transitional version, as well as the `frameset` element (used instead of `body`) and the `frame` element.

Frameset versus transitional

In addition to the above transitional differences, the frameset specifications (whether XHTML 1.0 or HTML 4.01) specify a different content model, with `frameset` replacing `body`, that contains either `frame` elements, or optionally `noframes` with a `body`.

Summary of specification versions

As this list demonstrates, the loose versions of the specification are maintained for legacy support. However, contrary to popular misconceptions, the move to XHTML does not imply a removal of this legacy support. Rather the X in XML stands for extensible and the W3C is modularizing the entire specification and opens it up to independent extensions. The primary achievement in the move from XHTML 1.0 to XHTML 1.1 is the modularization of the entire specification. The strict version of HTML is deployed in XHTML 1.1 through a set of modular extensions to the base XHTML 1.1 specification. Likewise, someone looking for the loose

(transitional) or frameset specifications will find similar extended XHTML 1.1 support (much of it is contained in the legacy or frame modules). Modularization also allows for separate features to develop on their own timetable. So for example, XHTML 1.1 will allow quicker migration to emerging XML standards such as MathML (a presentational and semantic math language based on XML) and XForms—a new highly advanced web-form technology to replace the existing HTML forms.
In summary, the HTML 4 specification primarily reined in all the various HTML implementations into a single clearly written specification based on SGML. XHTML 1.0, ported this specification, as is, to the new XML-defined specification. Next, XHTML 1.1 takes advantage of the extensible nature of XML and modularizes the whole specification. XHTML 2.0 was intended to be the first step in adding new features to the specification in a standards-body-based approach.

WHATWG HTML versus HTML5

The HTML Living Standard, which is developed by WHATWG, is the official version, while W3C HTML5 is no longer separate from WHATWG.

WYSIWYG editors

There are some WYSIWYG editors (*what you see is what you get*), in which the user lays out everything as it is to appear in the HTML document using a graphical user interface (GUI), often similar to word processors. The editor renders the document rather than showing the code, so authors do not require extensive knowledge of HTML.
The WYSIWYG editing model has been criticized,[97][98] primarily because of the low quality of the generated code; there are voices advocating a change to the WYSIWYM model (*what you see is what you mean*).
WYSIWYG editors remain a controversial topic because of their perceived flaws such as:

- Relying mainly on the layout as opposed to meaning, often using markup that does not convey the intended meaning but simply copies the layout.[99]
- Often producing extremely verbose and redundant code that fails to make use of the cascading nature of HTML and CSS.
- Often producing ungrammatical markup, called tag soup or semantically incorrect markup (such as `<em>` for italics).
- As a great deal of the information in HTML documents is not in the layout, the model has been criticized for its "what you see is all you get"-nature.[100]

See also

- Breadcrumb navigation
- Cellpadding

- Comparison of HTML parsers
- Dynamic web page
- HTML Application
- HTML character references
- List of document markup languages
- List of XML and HTML character entity references
- Microdata (HTML)
- Microformat
- Polyglot markup
- Semantic HTML
- W3C (X)HTML Validator
- Web colors

Notes

0. Even though HTML can be run in a browser, it is not viewed as a programming language in programming language discourse.[2]

References

0. "W3C Html" (https://www.w3.org/html/).
0. Hermans, Felienne; Schlesinger, Ari (2024-10-17). "A Case for Feminism in Programming Language Design" (https://dl.acm.org/doi/10.1145/3689492.3689809). *Proceedings of the 2024 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*. ACM. pp. 205–222. doi:10.1145/3689492.3689809 (https://doi.org/10.1145%2F3689492.3689809). ISBN 979-8-4007-1215-9. `{{cite book}}: | journal= ignored (help)`
0. "HTML 4.0 Specification — W3C Recommendation — Conformance: requirements and recommendations" (https://www.w3.org/TR/REC-html40-971218/conform.html#deprecated). World Wide Web Consortium. December 18, 1997. Archived (https://web.archive.org/web/20150705040855/http://www.w3.org/TR/REC-html40-971218/conform.html) from the original on July 5, 2015. Retrieved July 6, 2015.
0. Tim Berners-Lee, "Information Management: A Proposal (https://www.w3.org/History/1989/proposal.html)". CERN (March 1989, May 1990). W3C.
0. Berners-Lee, Tim. "Intended Uses" (https://www.w3.org/DesignIssues/Uses.html). *W3C*.

0. "Tags used in HTML" (http://info.cern.ch/hypertext/WWW/MarkUp/Tags.html). *info.cern.ch*. October 1991. Retrieved 2 March 2023.
0. "Tags used in HTML" (https://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/Tags.html). World Wide Web Consortium. November 3, 1992. Archived (https://web.archive.org/web/20100131184344/http://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/Tags.html) from the original on January 31, 2010. Retrieved November 16, 2008.
0. Berners-Lee, Tim (October 29, 1991). "Re: status. Re: X11 BROWSER for WWW" (http://lists.w3.org/Archives/Public/www-talk/1991SepOct/0003.html). World Wide Web Consortium. Archived (https://web.archive.org/web/20070524045009/http://lists.w3.org:80/Archives/Public/www-talk/1991SepOct/0003.html) from the original on May 24, 2007. Retrieved April 8, 2007.
0. "Index of the HTML 4 elements" (https://www.w3.org/TR/1999/REC-html401-19991224/index/elements). World Wide Web Consortium. December 24, 1999. Archived (https://web.archive.org/web/20070505172415/https://www.w3.org/TR/1999/REC-html401-19991224/index/elements) from the original on May 5, 2007. Retrieved April 8, 2007.
0. Berners-Lee, Tim (December 9, 1991). "Re: SGML/HTML docs, X Browser" (http://lists.w3.org/Archives/Public/www-talk/1991NovDec/0020.html). *w3*. Archived (https://web.archive.org/web/20071222060359/http://lists.w3.org/Archives/Public/www-talk/1991NovDec/0020.html) from the original on December 22, 2007. Retrieved June 16, 2007. "SGML is very general. HTML is a specific application of the SGML basic syntax applied to hypertext documents with simple structure."
0. Berners-Lee, Tim; Connolly, Daniel (June 1993). "Hypertext Markup Language (HTML): A Representation of Textual Information and MetaInformation for Retrieval and Interchange" (https://www.w3.org/MarkUp/draft-ietf-iiir-html-01.txt). *w3*. Archived (https://web.archive.org/web/20170103041713/https://www.w3.org/MarkUp/draft-ietf-iiir-html-01.txt) from the original on January 3, 2017. Retrieved January 4, 2017.
0. Raggett, Dave. "A Review of the HTML+ Document Format" (http://www.w3.org/MarkUp/htmlplus_paper/htmlplus.html). *w3*. Archived (https://web.archive.org/web/20000229205146/http://www.w3.org/MarkUp/htmlplus_paper/htmlplus.html) from the original on February 29, 2000. Retrieved May 22, 2020. "The hypertext markup language HTML was developed as a simple non-proprietary delivery format for global hypertext. HTML+ is a set of modular extensions to HTML and has been developed in response to a growing understanding of the needs of information providers. These extensions include text flow around floating figures, fill-out forms, tables, and mathematical equations."

0. Berners-Lee, Tim; Connolly, Daniel W. (November 1995). *Hypertext Markup Language - 2.0* (https://datatracker.ietf.org/doc/html/rfc1866). Network Working Group. doi:10.17487/RFC1866 (https://doi.org/10.17487%2FRFC1866). RFC 1866 (https://datatracker.ietf.org/doc/html/rfc1866). *Historic.* Obsoleted by RFC 2854 (https://datatracker.ietf.org/doc/html/rfc2854). "This document thus defines an HTML 2.0 (to distinguish it from the previous informal specifications). Future (generally upwardly compatible) versions of HTML with new features will be released with higher version numbers."
0. Raggett, Dave (1998). *Raggett on HTML 4* (https://web.archive.org/web/20070809234115/https://www.w3.org/People/Raggett/book4/ch02.html). Archived from the original (https://www.w3.org/People/Raggett/book4/ch02.html) on August 9, 2007. Retrieved July 9, 2007.
0. "HTML5 – Hypertext Markup Language – 5.0" (https://www.w3.org/2014/10/html5-rec.html.en). Internet Engineering Task Force. 28 October 2014. Archived (https://web.archive.org/web/20141028233921/https://www.w3.org/2014/10/html5-rec.html.en) from the original on October 28, 2014. Retrieved November 25, 2014. "This document recommends HTML 5.0 after completion."
0. "HTML 3.2 Reference Specification" (https://www.w3.org/TR/REC-html32). World Wide Web Consortium. January 14, 1997. Retrieved November 16, 2008.
0. "IETF HTML WG" (https://www.w3.org/MarkUp/HTML-WG/). Retrieved June 16, 2007. "Note: This working group is closed"
0. Engelfriet, Arnoud. "Introduction to Wilbur" (http://htmlhelp.com/reference/wilbur/intro.html). *htmlhelp.com*. Retrieved June 16, 2007.
0. "HTML 4.0 Specification" (https://www.w3.org/TR/REC-html40-971218/). World Wide Web Consortium. December 18, 1997. Retrieved November 16, 2008.
0. "HTML 4 – 4 Conformance: requirements and recommendations" (https://www.w3.org/TR/html4/conform.html#h-4.2). Retrieved December 30, 2009.
0. "HTML 4.0 Specification" (https://www.w3.org/TR/1998/REC-html40-19980424/). World Wide Web Consortium. April 24, 1998. Retrieved November 16, 2008.
0. "HTML 4.01 Specification" (https://www.w3.org/TR/html401/). World Wide Web Consortium. December 24, 1999. Retrieved November 16, 2008.
0. "HTML 4 Errata" (https://www.w3.org/MarkUp/html4-updates/errata). W3C. Retrieved March 2, 2023.
0. ISO (2000). "ISO/IEC 15445:2000 – Information technology – Document description and processing languages – HyperText Markup Language (HTML)" (https://www.iso.org/standard/27688.html). Retrieved March 1, 2023.
0. "ISO/IEC 15445:2000(E) ISO-HTML" (https://www.scss.tcd.ie/misc/15445/15445.HTML). *www.scss.tcd.ie*. Geneva, CH: ISO/IEC. May 15, 2000. Retrieved March 1, 2023.
0. "HTML5: A vocabulary and associated APIs for HTML and XHTML" (https://www.w3.org/TR/2014/REC-html5-20141028/). World Wide Web Consortium. 28 October 2014. Retrieved 31 October 2014.

0. "Open Web Platform Milestone Achieved with HTML5 Recommendation" (https://www.w3.org/2014/10/html5-rec.html.en) (Press release). World Wide Web Consortium. 28 October 2014. Retrieved 31 October 2014.
0. "HTML 5.1" (https://www.w3.org/TR/2016/REC-html51-20161101/). World Wide Web Consortium. 1 November 2016. Retrieved 6 January 2017.
0. "HTML 5.1 is a W3C Recommendation" (https://www.w3.org/blog/news/archives/5932). World Wide Web Consortium. 1 November 2016. Retrieved 6 January 2017.
0. Philippe le Hegaret (17 November 2016). "HTML 5.1 is the gold standard" (https://www.w3.org/blog/2016/11/html-5-1-is-the-gold-standard/). World Wide Web Consortium. Retrieved 6 January 2017.
0. "HTML 5.2" (https://www.w3.org/TR/2017/REC-html52-20171214/). World Wide Web Consortium. 14 December 2017. Retrieved 15 December 2017.
0. "HTML 5.2 is now a W3C Recommendation" (https://www.w3.org/blog/news/archives/6696). World Wide Web Consortium. 14 December 2017. Retrieved 15 December 2017.
0. Charles McCathie Nevile (14 December 2017). "HTML 5.2 is done, HTML 5.3 is coming" (https://www.w3.org/blog/2017/12/html-5-2-is-done-html-5-3-is-coming/). World Wide Web Consortium. Retrieved 15 December 2017.
0. Connolly, Daniel (6 June 1992). "MIME as a hypertext architecture" (http://lists.w3.org/Archives/Public/www-talk/1992MayJun/0020.html). CERN. Retrieved 24 October 2010.
0. Connolly, Daniel (15 July 1992). "HTML DTD enclosed" (http://lists.w3.org/Archives/Public/www-talk/1992JulAug/0020.html). CERN. Retrieved 24 October 2010.
0. Connolly, Daniel (18 August 1992). "document type declaration subset for Hyper Text Markup Language as defined by the World Wide Web project" (https://web.archive.org/web/20120314055308/http://lost-contact.mit.edu/afs/cern.ch/w3.org/www/Frame/fminit2.0/html.dtd). CERN. Archived from the original (http://lost-contact.mit.edu/afs/cern.ch/w3.org/www/Frame/fminit2.0/html.dtd) on 14 March 2012. Retrieved 24 October 2010.
0. Connolly, Daniel (24 November 1992). "Document Type Definition for the Hyper Text Markup Language as used by the World Wide Web application" (https://web.archive.org/web/20120118155040/http://lost-contact.mit.edu/afs/cern.ch/w3.org/www/MarkUp/Connolly/921125/archive.sh#html.dtd). CERN. Archived from the original (http://lost-contact.mit.edu/afs/cern.ch/w3.org/www/MarkUp/Connolly/921125/archive.sh#html.dtd) on 18 January 2012. Retrieved 24 October 2010. See section "Revision History"
0. Berners-Lee, Tim; Connolly, Daniel (June 1993). "Hyper Text Markup Language (HTML) Internet-Draft version 1.1" (http://tools.ietf.org/html/draft-ietf-iiir-html-00). IETF IIIR Working Group. Retrieved 18 September 2010.

0. Berners-Lee, Tim; Connolly, Daniel (June 1993). "Hypertext Markup Language (HTML) Internet-Draft version 1.2" (https://www.w3.org/MarkUp/draft-ietf-iiir-html-01.txt). IETF IIIR Working Group. Retrieved 18 September 2010.
0. Raggett, Dave (1993-11-08). "History for draft-raggett-www-html-00" (https://datatracker.ietf.org/doc/draft-raggett-www-html/history/). *IETF Datatracker*. Retrieved 2019-11-18.
0. Berners-Lee, Tim; Connolly, Daniel (28 November 1994). "HyperText Markup Language Specification – 2.0 INTERNET DRAFT" (http://tools.ietf.org/html/draft-ietf-html-spec-00). *Internet Engineering Task Force*. Retrieved 24 October 2010.
0. Connolly, Daniel W. (1995-05-16). "Hypertext Markup Language – 2.0" (https://tools.ietf.org/html/draft-ietf-html-spec-02#section-1.1). *tools.ietf.org*. Retrieved 2019-11-18.
0. Berners-Lee, Tim; Connolly, Daniel W. (November 1995). *Hypertext Markup Language - 2.0* (https://datatracker.ietf.org/doc/html/rfc1866). Network Working Group. doi:10.17487/RFC1866 (https://doi.org/10.17487%2FRFC1866). RFC 1866 (https://datatracker.ietf.org/doc/html/rfc1866). *Historic.* Obsoleted by RFC 2854 (https://datatracker.ietf.org/doc/html/rfc2854).
0. "HTML 3.0 Draft (Expired!) Materials" (https://www.w3.org/MarkUp/html3/). World Wide Web Consortium. December 21, 1995. Retrieved November 16, 2008.
0. "HyperText Markup Language Specification Version 3.0" (https://www.w3.org/MarkUp/html3/CoverPage). Retrieved June 16, 2007.
0. Raggett, Dave (28 March 1995). "HyperText Markup Language Specification Version 3.0" (https://www.w3.org/People/Raggett/html3/html3.txt). *HTML 3.0 Internet Draft Expires in six months*. World Wide Web Consortium. Retrieved 17 June 2010.
0. Bowers, N. (1998). "Weblint: just another perl hack" (https://www.usenix.org/publications/library/proceedings/usenix98/freenix/bowers.pdf) (PDF). *1998 USENIX Annual Technical Conference (USENIX ATC 98)*.
0. Lie, Håkon Wium; Bos, Bert (April 1997). *Cascading style sheets: designing for the Web* (https://archive.org/details/cascadingstylesh00lieh). Addison Wesley Longman. p. 263 (https://archive.org/details/cascadingstylesh00lieh/page/263). ISBN 978-0-201-41998-6. Retrieved 9 June 2010.
0. "HTML5" (https://www.w3.org/TR/html5/). World Wide Web Consortium. June 10, 2008. Retrieved November 16, 2008.
0. "HTML5, one vocabulary, two serializations" (https://www.w3.org/blog/2008/01/html5-is-html-and-xml/). 15 January 2008. Retrieved February 25, 2009.
0. "W3C Confirms May 2011 for HTML5 Last Call, Targets 2014 for HTML5 Standard" (https://www.w3.org/2011/02/htmlwg-pr.html). World Wide Web Consortium. 14 February 2011. Retrieved 18 February 2011.

0. Hickson, Ian (January 19, 2011). "HTML Is the New HTML5" (https://web.archive.org/web/20191006023430/https://blog.whatwg.org/html-is-the-new-html5). *The WHATWG Blog*. Archived from the original (https://blog.whatwg.org/html-is-the-new-html5) on 6 October 2019. Retrieved 21 January 2011.
0. Grannell, Craig (July 23, 2012). "HTML5 gets the splits" (https://web.archive.org/web/20120725214739/http://www.netmagazine.com/news/html5-gets-splits-122102). Net magazine. Archived from the original (http://www.netmagazine.com/news/html5-gets-splits-122102) on Jul 25, 2012. Retrieved 23 July 2012.
0. "HTML5" (https://www.w3.org/TR/2012/CR-html5-20121217/). W3C. 2012-12-17. Retrieved 2013-06-15.
0. "When Will HTML5 Be Finished?" (https://wiki.whatwg.org/wiki/FAQ#What.27s_this_I_hear_about_2022.3F). *FAQ*. WHAT Working Group. Retrieved 29 November 2009.
0. "Call for Review: HTML5 Proposed Recommendation Published W3C News" (https://www.w3.org/blog/news/archives/4074). W3C. 2014-09-16. Retrieved 2014-09-27.
0. "Open Web Platform Milestone Achieved with HTML5 Recommendation" (https://www.w3.org/2014/10/html5-rec.html.en). W3C. 28 October 2014. Retrieved 29 October 2014.
0. "HTML5 specification finalized, squabbling over specs continues" (https://arstechnica.com/information-technology/2014/10/html5-specification-finalized-squabbling-over-who-writes-the-specs-continues/). *Ars Technica*. 2014-10-29. Retrieved 2014-10-29.
0. "HTML vs XML syntax" (https://html.spec.whatwg.org/multipage/introduction.html#html-vs-xhtml). WHATWG. Retrieved 22 March 2023.
0. "XHTML 1.0: The Extensible HyperText Markup Language (Second Edition)" (https://www.w3.org/TR/xhtml1/). World Wide Web Consortium. January 26, 2000. Retrieved November 16, 2008.
0. "XHTML 1.1 – Module-based XHTML — Second Edition" (https://www.w3.org/TR/xhtml11/). World Wide Web Consortium. February 16, 2007. Retrieved November 16, 2008.
0. "Modularization of XHTML" (https://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/). *W3C*. Retrieved 2017-01-04.
0. "XHTM 2.0" (https://www.w3.org/TR/xhtml2/). World Wide Web Consortium. July 26, 2006. Retrieved November 16, 2008.
0. "XHTML 2 Working Group Expected to Stop Work End of 2009, W3C to Increase Resources on HTML5" (https://www.w3.org/News/2009#item119). World Wide Web Consortium. July 17, 2009. Retrieved November 16, 2008.
0. "W3C XHTML FAQ" (https://www.w3.org/2009/06/xhtml-faq.html).

0. Jaffe, Jeff (28 May 2019). "W3C and WHATWG to Work Together to Advance the Open Web Platform" (https://www.w3.org/blog/2019/05/w3c-and-whatwg-to-work-together-to-advance-the-open-web-platform/). *W3C Blog*. Archived (https://web.archive.org/web/20190529021122/https://www.w3.org/blog/2019/05/w3c-and-whatwg-to-work-together-to-advance-the-open-web-platform/) from the original on 29 May 2019. Retrieved 29 May 2019.
0. "W3C and the WHATWG Signed an Agreement to Collaborate on a Single Version of HTML and DOM" (https://www.w3.org/html/). *W3C*. 28 May 2019. Archived (https://web.archive.org/web/20190529012655/https://www.w3.org/html/) from the original on 29 May 2019. Retrieved 29 May 2019.
0. "Memorandum of Understanding Between W3C and WHATWG" (https://www.w3.org/2019/04/WHATWG-W3C-MOU.html). *W3C*. 28 May 2019. Archived (https://web.archive.org/web/20190529012854/https://www.w3.org/2019/04/WHATWG-W3C-MOU.html) from the original on 29 May 2019. Retrieved 29 May 2019.
0. Cimpanu, Catalin (29 May 2019). "Browser vendors Win War with W3C over HTML and DOM standards" (https://web.archive.org/web/20190529021959/https://www.zdnet.com/article/browser-vendors-win-war-with-w3c-over-html-and-dom-standards/). *ZDNet*. Archived from the original (https://www.zdnet.com/article/browser-vendors-win-war-with-w3c-over-html-and-dom-standards/) on 29 May 2019. Retrieved 29 May 2019.
0. "W3C – WHATWG Wiki" (https://web.archive.org/web/20190529013834/https://wiki.whatwg.org/wiki/W3C). *WHATWG Wiki*. Archived from the original (https://wiki.whatwg.org/wiki/W3C) on 29 May 2019. Retrieved 29 May 2019.
0. Shankland, Stephen (July 9, 2009). "An epitaph for the Web standard, XHTML 2" (https://www.cnet.com/news/an-epitaph-for-the-web-standard-xhtml-2/). *CNET*. CBS INTERACTIVE INC.
0. Activating Browser Modes with Doctype (https://hsivonen.iki.fi/doctype/). Hsivonen.iki.fi. Retrieved on 2012-02-16.
0. "HTML Elements" (https://www.w3schools.com/html/html_elements.asp). w3schools. Retrieved 16 March 2015.
0. "CSS Introduction" (https://www.w3schools.com/css/css_intro.asp). W3schools. Retrieved 16 March 2015.
0. "On SGML and HTML" (https://www.w3.org/TR/html401/intro/sgmltut.html#h-3.2.2). World Wide Web Consortium. Retrieved November 16, 2008.
0. "XHTML 1.0 – Differences with HTML 4" (https://www.w3.org/TR/xhtml1/diffs.html#h-4.4). World Wide Web Consortium. Retrieved November 16, 2008.
0. Korpela, Jukka (July 6, 1998). "Why attribute values should always be quoted in HTML" (https://jkorpela.fi/qattr.html). Cs.tut.fi. Retrieved November 16, 2008.

0. "Objects, Images, and Applets in HTML documents" (https://www.w3.org/TR/1999/REC-html401-19991224/struct/objects.html#adef-ismap). World Wide Web Consortium. December 24, 1999. Retrieved November 16, 2008.
0. "H56: Using the dir attribute on an inline element to resolve problems with nested directional runs" (https://www.w3.org/TR/WCAG-TECHS/H56.html). *Techniques for WCAG 2.0*. W3C. Retrieved 18 September 2010.
0. "Character Entity Reference Chart" (https://dev.w3.org/html5/html-author/charref). World Wide Web Consortium. October 24, 2012.
0. "The Named Character Reference ' " (https://www.w3.org/TR/xhtml1/#C_16). World Wide Web Consortium. January 26, 2000.
0. "*The Unicode Standard*: A Technical Introduction" (https://www.unicode.org/standard/principles.html). Unicode. Retrieved 2010-03-16.
0. "The HTML syntax" (https://www.w3.org/TR/html/syntax.html#doctype-syntax). *HTML Standard*. Retrieved 2013-08-19.
0. "HTML 4 Frameset Document Type Definition" (https://www.w3.org/TR/html401/sgml/framesetdtd.html). *W3C*. Retrieved 2021-12-25.
0. Berners-Lee, Tim; Fischetti, Mark (2000). *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor* (https://archive.org/details/weavingweborigin00bern_0). San Francisco: Harper. ISBN 978-0-06-251587-2.
0. Raggett, Dave (2002). "Adding a touch of style" (https://www.w3.org/MarkUp/Guide/Style.html). W3C. Retrieved October 2, 2009. This article notes that presentational HTML markup may be useful when targeting browsers "before Netscape 4.0 and Internet Explorer 4.0". See the list of web browsers to confirm that these were both released in 1997.
0. Berners-Lee, Tim; Hendler, James; Lassila, Ora (May 1, 2001). "The Semantic Web" (http://www.scientificamerican.com/article.cfm?id=the-semantic-web). *Scientific American*. Retrieved October 2, 2009.
0. Nigel Shadbolt, Wendy Hall and Tim Berners-Lee (2006). "The Semantic Web Revisited" (https://web.archive.org/web/20130320130521/http://eprints.soton.ac.uk/262614/1/Semantic_Web_Revisted.pdf) (PDF). IEEE Intelligent Systems. Archived from the original (http://eprints.ecs.soton.ac.uk/12614/1/Semantic_Web_Revisted.pdf) (PDF) on March 20, 2013. Retrieved October 2, 2009.
0. "HTML: The Living Standard" (https://html.spec.whatwg.org/dev/introduction.html#restrictions-on-content-models-and-on-attribute-values). *WHATWG*. Retrieved 27 September 2018.

0. "XHTML 1.0 The Extensible HyperText Markup Language (Second Edition)" (https://www.w3.org/TR/xhtml1/#media). World Wide Web Consortium. 2002 [2000]. Retrieved December 7, 2008. "XHTML Documents which follow the guidelines set forth in Appendix C, "HTML Compatibility Guidelines" may be labeled with the Internet Media Type "text/html" [RFC2854], as they are compatible with most HTML browsers. Those documents, and any other document conforming to this specification, may also be labeled with the Internet Media Type "application/xhtml+xml" as defined in [RFC3236]."
0. S. Bradner (March 1997). *Key words for use in RFCs to Indicate Requirement Levels* (https://datatracker.ietf.org/doc/html/rfc2119). Network Working Group. doi:10.17487/RFC2119 (https://doi.org/10.17487%2FRFC2119). BCP 14. RFC 2119 (https://datatracker.ietf.org/doc/html/rfc2119). *Best Current Practice 14.* Updated by RFC 8174 (https://datatracker.ietf.org/doc/html/rfc8174). "3. SHOULD This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course."
0. "XHTML 1.1 – Module-based XHTML — Second Edition" (https://www.w3.org/TR/xhtml11/conformance.html#strict). World Wide Web Consortium. 2007. Retrieved December 7, 2008. "XHTML 1.1 documents SHOULD be labeled with the Internet Media Type text/html as defined in [RFC2854] or application/xhtml+xml as defined in [RFC3236]."
0. "Naming Files, Paths, and Namespaces" (https://msdn.microsoft.com/en-us/library/windows/desktop/aa365247%28v=vs.85%29.aspx?f=255&MSPPError=-2147217396). Microsoft. Retrieved 16 March 2015.
0. HTML Design Constraints (https://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/HTMLConstraints.html), W3C Archives
0. WWW: BTB – HTML (http://ei.cs.vt.edu/~wwwbtb/book/chap13/who.html), Pris Sears
0. *HTML Standard - The br Element* (https://html.spec.whatwg.org/multipage/text-level-semantics.html#the-br-element), WHATWG
0. Sauer, C.: WYSIWIKI – Questioning WYSIWYG in the Internet Age. In: Wikimania (2006)
0. Spiesser, J., Kitchen, L.: Optimization of HTML automatically generated by WYSIWYG programs. In: 13th International Conference on World Wide Web, pp. 355—364. WWW '04. ACM, New York, NY (New York, NY, U.S., May 17–20, 2004)
0. XHTML Reference: blockquote (http://xhtml.com/en/xhtml/reference/blockquote/) Archived (https://web.archive.org/web/20100325160356/http://xhtml.com/en/xhtml/reference/blockquote/) 2010-03-25 at the Wayback Machine. Xhtml.com. Retrieved on 2012-02-16.
0. Doug Engelbart's INVISIBLE REVOLUTION (http://www.invisiblerevolution.net/). Invisiblerevolution.net. Retrieved on 2012-02-16.

External links

- WHATWG's HTML Living Standard (https://html.spec.whatwg.org/multipage/)
- Dave Raggett's Introduction to HTML (https://www.w3.org/MarkUp/Guide/)
- Tim Berners-Lee Gives the Web a New Definition (https://web.archive.org/web/20110412130543/http://computemagazine.com/man-who-invented-world-wide-web-gives-new-definition) (archived 12 April 2011)
- List of all HTML elements from all major versions (https://meiert.com/en/indices/html-elements/)
- HTML Entities (https://www.w3schools.com/html/html_entities.asp)
- Sean B. Palmer. "Early History of HTML – 1990 to 1992" (http://infomesh.net/html/history/early/). *Infomesh*. Retrieved 2022-04-13. (Timeframe: 1980–1995)