Team JOKE

Contents

Description

PostScript (PS) is a page description language in the electronic publishing and desktop publishing business. It is a dynamically typed, concatenative programming language.

It was created at Adobe Systems by John Warnock, Charles Geschke, Doug Brotz, Ed Taft and Bill Paxton from 1982 to 1984.

It is a printing file format with a dynamic type system.

History

- It was created at Adobe Systems by John Warnock, Charles Geschke, Doug Brotz, Ed Taft and Bill Paxton from **1982** to **1984**.
- The concepts of the PostScript language were seeded in **1976**.
- In **1978** John Gaffney and Martin Newell then at Xerox PARC wrote J & M or JaM which was used for **VLSI design** and the investigation of type and graphics printing.
- In **March 1985**, the Apple LaserWriter was the first printer to ship with PostScript, sparking the desktop publishing (DTP) revolution in the mid-1980s.
- For a time an interpreter (**Raster Image Processor[RIP]**) for the PostScript language was a common component of laser printers, into the **1990s.**
- By **2001**, few lower-end printer models came with support for PostScript, **largely due to growing competition** from much cheaper **non-PostScript** inkjet printers.

History

There are **three versions** of Postscript being created.

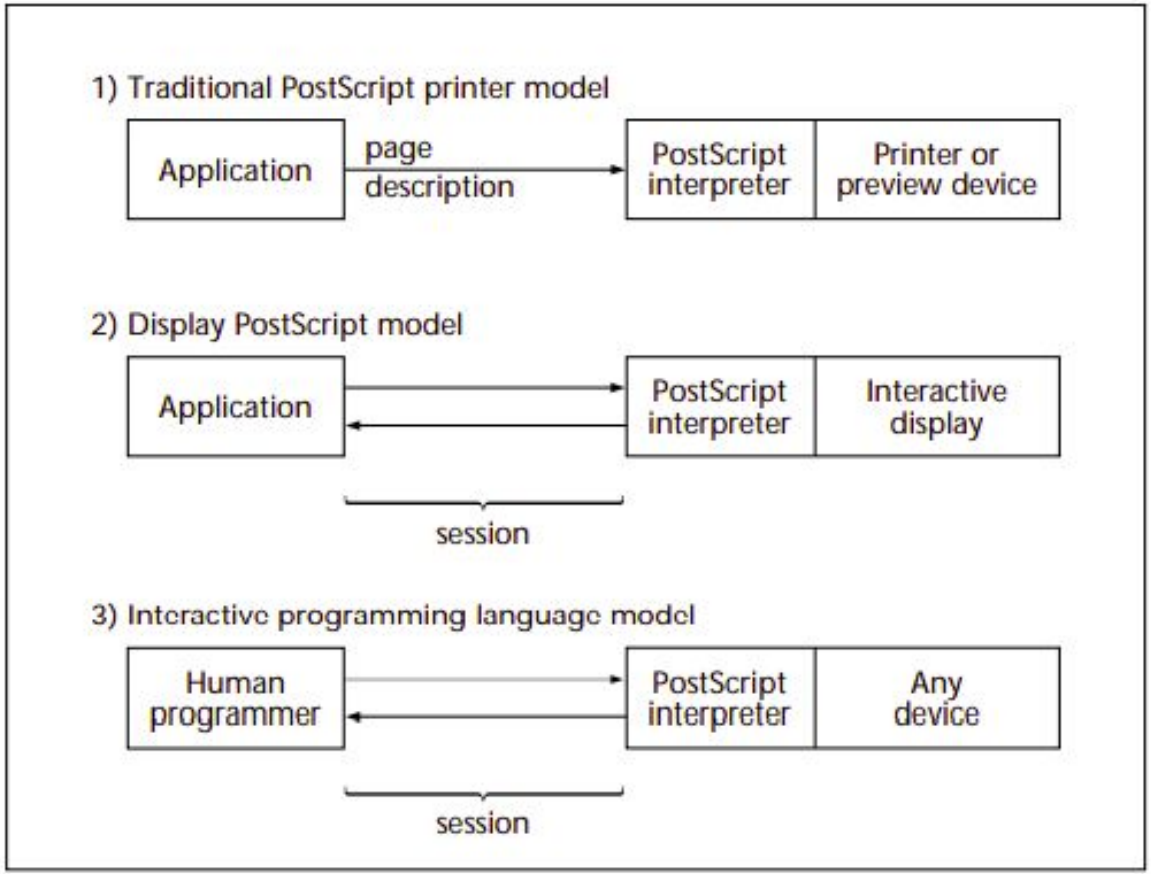
- **PostScript Level 1 (**suffix Level 1 was added when Level 2 was introduced**)**
  - The **first version** of the PostScript language was released to the market in 1984.
- **PostScript Level 2**
  - Was introduced in **1991**, and included several improvements.
    - Improved speed and reliability
    - Support for in-RIP separations
    - Image decompression, & etc.
- **PostScript 3**
  - PostScript 3 (Adobe dropped the "level" terminology in favor of simple versioning) came at the end of **1997.**
  - Widely used for magazine production, through the introduction of smooth shading operations with up to **4096 shades of grey** (rather than the 256 available in PostScript Level 2)
  - Introduced better color handling and new filters.

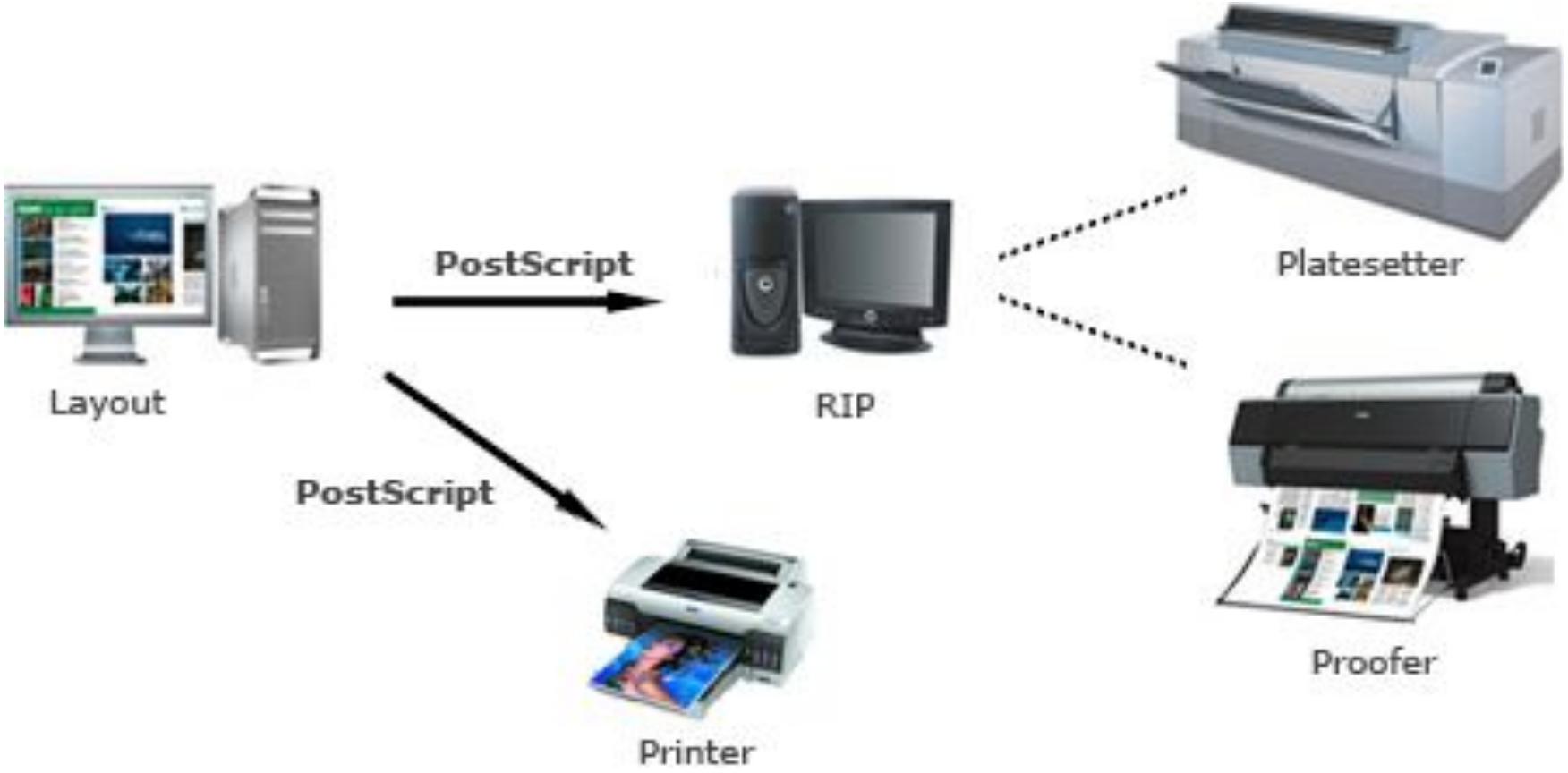
Domain

PostScript is a Turing-complete Language

In practice, PostScript is mainly used as a page description language.

PostScript is designed to convey virtually any desired page to any brand of printer at the time


1) Traditional PostScript printer model
Application
page description
PostScript interpreter
Printer or preview device
2) Display PostScript model
Application
PostScript interpreter
Interactive display
session
3) Interactive programming language model
Human programmer
PostScript interpreter
Any device
session

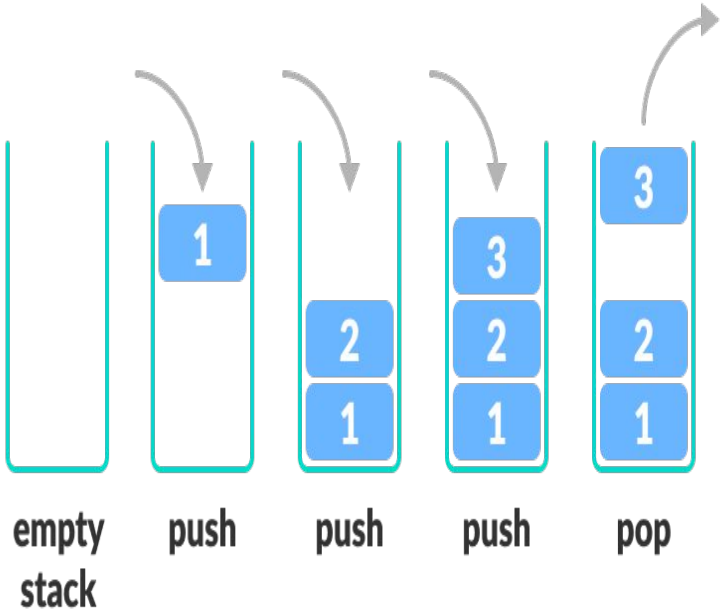

PostScript
PostScript
Layout
RIP
Platesetter
Proofer
Printer

Paradigm

PostScript is a multi-paradigm language

Stack Based

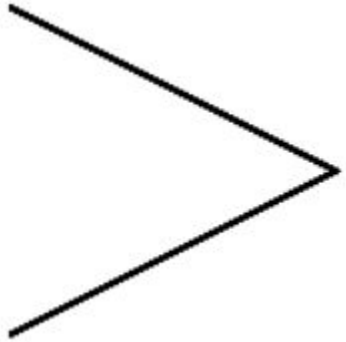
- relies on a **stack machine model** for passing parameters
- very easy for computers to evaluate and generate
- **does not** require large amount of memory and computer power to evaluate and generate



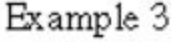
Procedural

- The program code is written as a **sequence of instructions**.
- User has to specify “what to do” and also “how to do” (step by step procedure)
- These instructions are executed in the sequential order.

```
newpath
100 200 moveto
200 250 lineto
100 300 lineto
2 setlinewidth
stroke
```



```
/Times-Roman findfont
12 scalefont
setfont
newpath
100 200 moveto
(Example 3) show
```



Features

- Data type
- Operators
- Data Structures
- Control Structures

Data Types

- Simple Objects
  - boolean
  - fontID
  - integer
  - real
  - mark
  - name
  - null
  - operator
  - Save
- Composite Objects
  - array
  - dictionary
  - file
  - gstate
  - packedarray
  - string

Operators

- Stack Operators
- Arithmetic and Mathematical Operators
- Array, Packed Array, Dictionary, and String Operators
- Relational, Boolean, and Bitwise Operators
- Control Operators
- Type, Attribute, and Conversion Operators

Stack Operators

- dup
- exch
- pop
- copy
- roll
- index
- mark
- clear
- count
- counttomark

Arithmetic and Mathematical Operators

- **add**, **sub**, **mul**, **div**, **idiv**, and **mod** are arithmetic operators of two arguments.
- **abs**, **neg**, **ceiling**, **floor**, **round**, and **truncate** are arithmetic operators of one argument.
- **sqrt**, **exp**, **ln**, **log**, **sin**, **cos**, and **atan** are mathematical and trigonometric functions.
- **rand**, **srand**, and **rrand** access a pseudo-random number generator.

Array, Packed Array, Dictionary, and String Operators

- get
- put
- copy
- length
- forall
- getinterval
- putinterval

The following operators apply only to arrays and (sometimes) packed arrays:

- **aload** and **astore**
- **setpacking** and **currentpacking**
- The array construction operators [and] combine to produce a new array object
- Packed array objects are always read-only, so the **put**, **putinterval**, and **astore** operations are not allowed on them.

The following operators apply only to dictionaries:

- **begin** and **end**
- **def** and **store**
- **countdictstack**, **cleardictstack**, and **dictstack**
- **known**
- **maxlength**
- **undef** (*Level 2*)
- **<<** and **>>** (*Level 2*)

The following operators apply only to strings:

- **search** and **anchorsearch**
- **token**

Relational, Boolean, and Bitwise Operators

The relational operators compare two operands and produce a boolean result indicating if the relation holds. Any two objects may be compared for equality

eq - equal
ne - not equal

numbers and strings may be compared by the inequality operators

gt - greater than
ge - greater than or equal to
le - less than or equal to
lt - less than

The boolean and bitwise operators (**and**, **or**, **xor**, **true**, **false**, and **not**) compute logical combinations of boolean operands or bitwise combinations of integer operands.

The bitwise shift operator **bitshift** applies only to integers.

Control Operators

- **if** and **ifelse**
- **exec**
- **for**, **repeat**, **loop**, **forall**, **pathforall** and **kshow**.
- **exit**
- **countexecstack** and **execstack**

Type, Attribute, and Conversion Operators

- **type**
- **xcheck**, **rcheck**, and **wcheck**
- **cvlit** and **cvx**
- **readonly**, **executeonly**, and **noaccess**
- **cvi** and **cvr**
- **cvn**
- **cvs** and **cvrs**

Data Structure

- The PostScript language provides a unique data structure known as a dictionary.
- The PostScript language also provides several other standard data structures: strings, arrays (of any PostScript objects, including other arrays), integers, reals, names (analogous to identifiers in other languages), and files (with standard file operations defined).
- The PostScript language is based on stacks.

Control Structure

The whole thing is stack oriented. Adding two numbers is done by pushing the two numbers on the stack, then invoking the add operator, which pops two numbers, adds them, and pushes the result:

```
2 2 add    % leaves 4 on the stack
```

Control structures work the same way. The *ifelse* operator takes three things from the stack: a boolean and two executable blocks. If the boolean is true, it executes the first blocks. If the boolean is true, it executes the first block and discards the second. If the boolean is false, it executes the second block and discards the first:

```
1 2 eq { (y) } { (n) } ifelse    % leaves "n" on the stack
```

The *for* operator is used for looping. It takes a start value, an increment, and end value, and an executable block as operands:

```
% Calls doSomething ten times with 1 through 10 as arguments
1 1 10 { doSomething } for
```