

# 프로그래밍 언어

컴퓨터정보과 권용광



## CONTENT

1. 컴퓨터 언어와 프로그래밍의 개념
2. 프로그래밍 과정과 컴퓨터 언어의 계층
3. 프로그래밍 언어의 발달 과정과 분류
4. 프로그래밍 코딩의 실제 예

- 컴퓨터 언어와 프로그래밍의 개념
- 프로그래밍 과정과 컴퓨터 언어의 계층 구조
- 프로그래밍 언어의 발달 과정과 분류
- 프로그래밍 코딩

# 컴퓨터 언어와 프로그래밍의 개념

Chapter 5 프로그래밍 언어



## 01

컴퓨터 언어와  
프로그래밍의 개념

### 컴퓨터 언어와 프로그래밍

#### 컴퓨터 언어

컴퓨터 시스템에서 작동하는 소프트웨어를 작성하기 위한 언어

컴퓨터와의 의사소통을 위한 약속

#### 프로그래밍 언어

프로그램 작성에 필요한 언어

정해진 규칙에 따라 정의된 인공적인 언어

(예) C, Basic, Java 등

## 01

컴퓨터 언어와  
프로그래밍의 개념

## 컴퓨터 언어와 프로그래밍

## 컴퓨터 언어 &amp; 컴퓨터 프로그래밍

C언어와 C언어의 프로그래밍은 다르다.

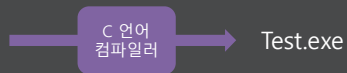
C언어 = 컴파일러. C언어를 프로그래밍 할 수 있게 하는 컴파일러 소프트웨어

C언어 프로그램 = C언어를 이용하여 개발된 소프트웨어

Test.cpp

```
#include <stdio.h>

main(){
    int a=10, b=20;
    int sum;
    sum= a+b;
    :
    :
}
```



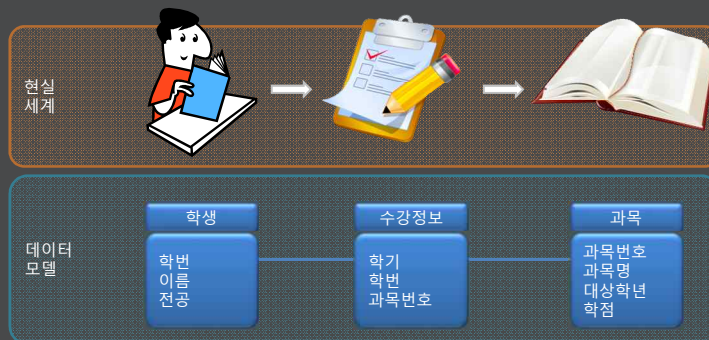
5

## 02

컴퓨터 언어와  
프로그래밍의 개념

## 소프트웨어 개발과 현실 세계

소프트웨어 개발 작업 → 실제 세상을 추상화 및 구체화하는 과정



6

## 02

컴퓨터 언어와  
프로그래밍의 개념

## 좋은 디자인의 조건

Good Design is simple.

Good Design is hard.

Good Design looks easy.

Good Design resembles nature.



7

## 02

컴퓨터 언어와  
프로그래밍의 개념

## Good 프로그래밍 언어의 조건

## Good 프로그래밍 언어란?

1. 언어의 개념이 명료하고, 문법적 구조가 일관성 있으며 단순해야 한다.
  - C →  $\text{sum} = a + b;$
  - COBOL → ADD A TO B GIVING C.
2. 좋은 프로그래밍 언어는 만들어지기가 어렵다.
3. 좋은 프로그래밍 언어는 배우기 쉬워야 한다.
4. 프로그래머의 생각을 구현하기가 용이해야 한다.
5. 프로그램의 호환성, 신뢰성, 모듈화, 효율성 등이 좋아야 한다.
6. 언어의 확장성이 우수해야 한다.
7. 검증이 쉬워야 한다.



8

## 02

컴퓨터 언어와  
프로그래밍의 개념

### Good 프로그래머란?

“잘 운영되는 소프트웨어를 빠른 시간 내에 만들 수 있는 능력을 갖는 사람”

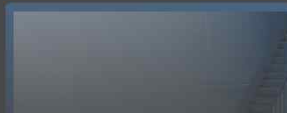


9

#### Section 02

## 프로그래밍 과정과 언어의 계층

Chapter 5 프로그래밍 언어



## 02

## 언어 처리 프로그램과 종류

프로그래밍 과정과  
컴퓨터 언어의 계층

종류	기능	설명
Compiler	고급언어 → 컴파일러 → 저급언어	번역기
Linker	여러 프로그램 → 링커 → 로드모듈	기계어로 된 여러 프로그램을 묶어서 실행 가능한 기계어(로드 모듈)로 번역
Loader	로드모듈 → 로더 → 주기억 장치	로더 모듈을 실행 가능한 기계어로 번역하여 주기억 장치에 적재
Preprocessor	고급언어 → 프리프로세서 → 고급언어	컴파일러 이전에, 특정 변수를 그것에 대응하는 정의된 문자열로 치환
Interpreter	고급언어 → 인터프리터 → 실행언어	고급 언어를 적당한 중간 코드까지만 번역해서 곧바로 실행하는 언어
Assembler	어셈블리어 → 어셈블러 → 기계어	원시 언어가 어셈블리 언어인 번역기



11

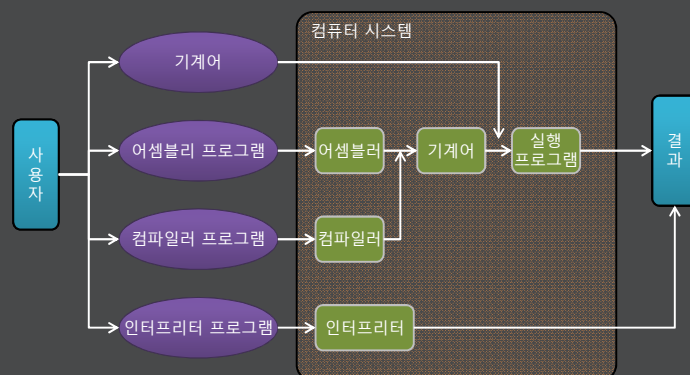
## 02

## 좁은 의미의 프로그래밍 과정

프로그래밍 과정과  
컴퓨터 언어의 계층

프로그램 작성 과정, 번역 과정, 실행 과정 등으로 구분.

프로그램 작성 과정은 프로그래밍 언어마다 작성(coding, 코딩) 규칙이 다름.



12

## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 기계어

## 기계어

컴퓨터가 직접 인식할 수 있는 언어

CPU의 명령과 2진코드를 1:1로 매칭. (따라서 CPU별로 명령어는 다름)

- (예) 0000 0000 → A + B  
0000 0001 → A - B  
0000 0010 → A의 값을 B로 옮겨라 ... 등등

[Quiz] 만일, 컴퓨터가 32bit 방식에서 64bit 방식으로 바뀌면...

## 어셈블리어

2진 부호를 니모닉화한 언어

- (예) 0000 0000 → A + B → ADD A, B  
0000 0001 → A - B → SUB A, B  
0000 0010 → A의 값을 B로 옮겨라 → MOV B, A

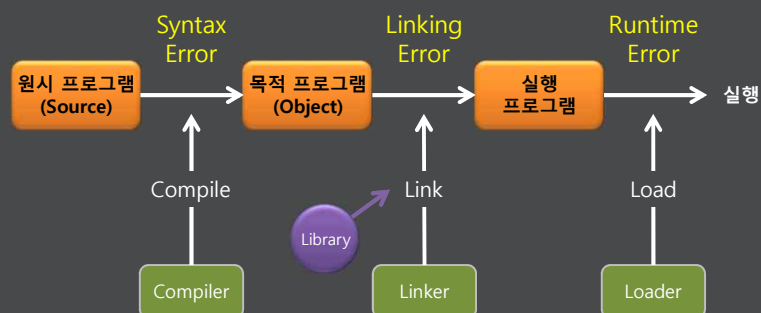


13

## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 컴파일 기법을 이용한 변환과 실행

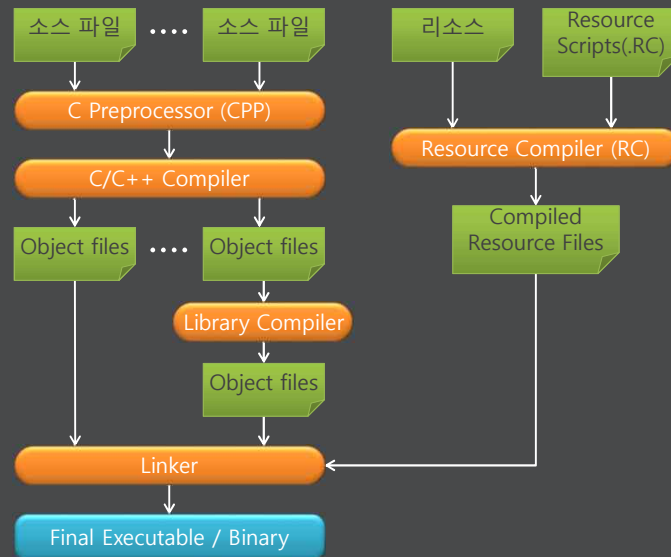


14

## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 컴파일 기법 : C 언어의 변환과 실행



## 02

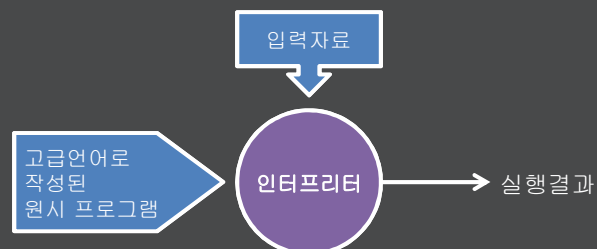
프로그래밍 과정과  
컴퓨터 언어의 계층

## 인터프리터 기법을 이용한 프로그램 변환 &amp; 실행

고급 언어를 적당한 중간 코드까지만 번역해서 곧바로 실행

인터프리터 : 원시코드 명령어들을 한번에 한 줄씩 읽어서 실행하는 프로그램.

프로그래밍 언어 : BASIC



장점 : 고급 프로그램을 대화식으로 직접 실행하여 결과를 알 수 있다.

따라서 개발단계 또는 교육과정에서 유용



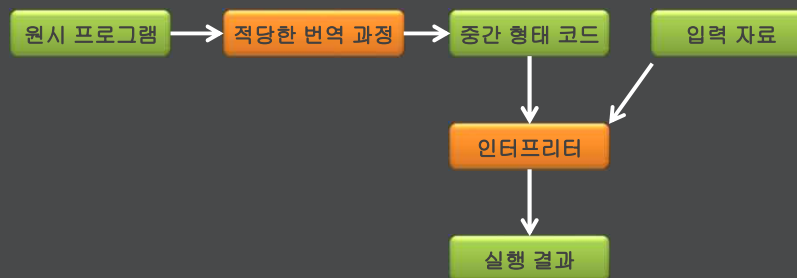
## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 혼합 기법을 이용한 프로그램 변환과 실행

혼합 기법 : 컴파일러와 인터프리터 기법을 혼용한 형태

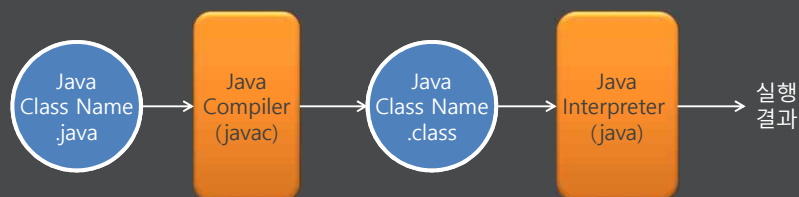
1. 프로그램을 좀더 실행시키기 쉬운 형태로 번역한 후,
2. 그 번역된 형태의 프로그램을 시뮬레이션으로 실행하는 구현 기법



## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

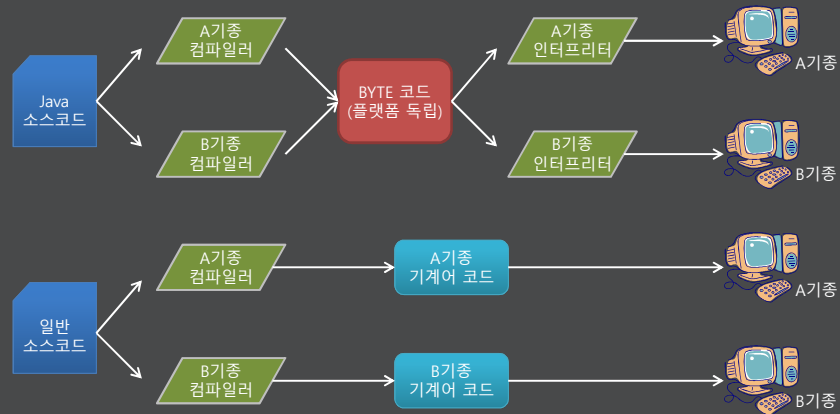
## 혼합 기법: Java의 프로그램 변환 &amp; 실행



## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 프로그래밍 언어와 자바 언어의 비교



19

## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 기계어와 어셈블리 언어의 실행 과정

## 기계어

## 최초의 컴퓨터 언어

컴퓨터가 번역 과정을 거치지 않고 직접 인식할 수 있는 유일한 언어

## 어셈블리 언어

기계어의 명령을 기호나 연상코드(mnemonic code)를 대신 사용해서 프로그램을 작성하기 때문에 '기호 언어(symbolic language)'라고도 한다.

종류	설명
기계어 (Machine Lang.)	<ul style="list-style-type: none"> <li>0/1의 조합 형태로 작성</li> <li>프로그램의 작성과 이해가 어렵다.</li> <li>프로그램의 유지 보수가 힘들다.</li> <li>컴퓨터에 관한 사전 지식이 필요</li> </ul>
어셈블리 (Assembly Lang.)	<ul style="list-style-type: none"> <li>기호어</li> <li>기계어의 단점 보완</li> <li>기계어에 비해 프로그램 작성이 용이</li> <li>기계어 보다 이해와 유지 보수가 편리</li> <li>어셈블리에 의한 번역이 필수</li> </ul>

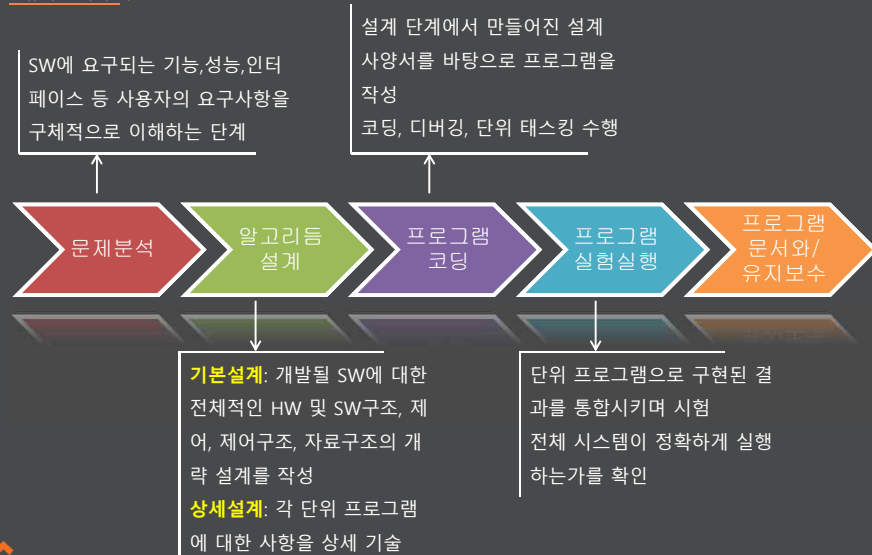


20

## 02

프로그래밍 과정과  
컴퓨터 언어의 계층

## 넓은 의미의 프로그래밍 과정과 소프트웨어 공학

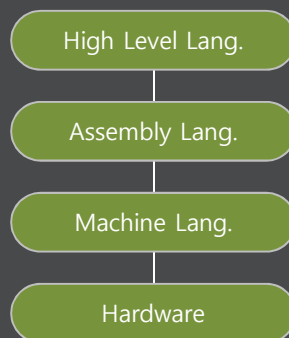


## 03

프로그래밍 과정과  
컴퓨터 언어의 계층

## 프로그래밍 언어의 계층과 프로그램 내장 방식

프로그래밍 언어의 계층



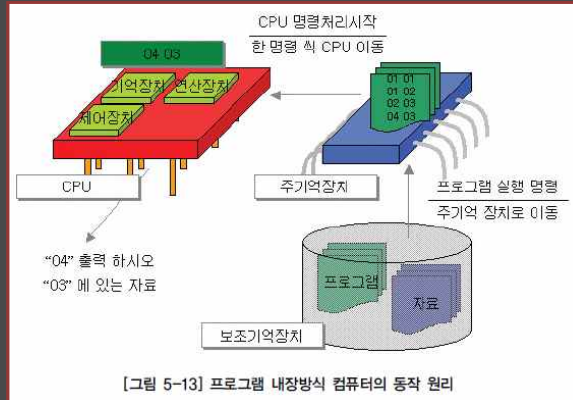
## 03

프로그래밍 과정과  
컴퓨터 언어의 계층

## 프로그래밍 언어의 계층과 프로그램 내장 방식

폰 노이만의 프로그램 내장방식에서의 번역기

1940년대 중반 미국 프린스턴 대학의 폰 노이만 (Von Neumann) 교수가 컴퓨터에  
계산 명령을 기억시키는 프로그램 내장방식 제시



sau

23

## Section 03

## 프로그래밍 언어 발달과정과 분류

Chapter 5 프로그래밍 언어



## 01

프로그래밍 언어  
발달과정과 분류

## 프로그래밍 언어의 초기와 발달과정

## 프로그래밍 언어의 발달 과정

년도	발달과정과 특징
1930~1940	프로그래밍 표기의 창조적인 방법들 소개 (Tuning, Flow diagram)
1950년대	기계어 / 어셈블리어 등장
1954~1957	Fortran (공학계산용)
1958~1960	알골60
1959~1960	Cobol (사무처리용)
1950년 후반	LISP (인공지능 분야 관련)
1964	BASIC (교육용, 가정용)
1971	C언어 (하드웨어에 근접된 언어)
1970년대	자료 추상화, 병행성, 증명 등의 메커니즘을 집중 시도
1980년대	Smalltalk 등 객체지향 언어 개발
1990년대	WEB 활성화. Java 등장

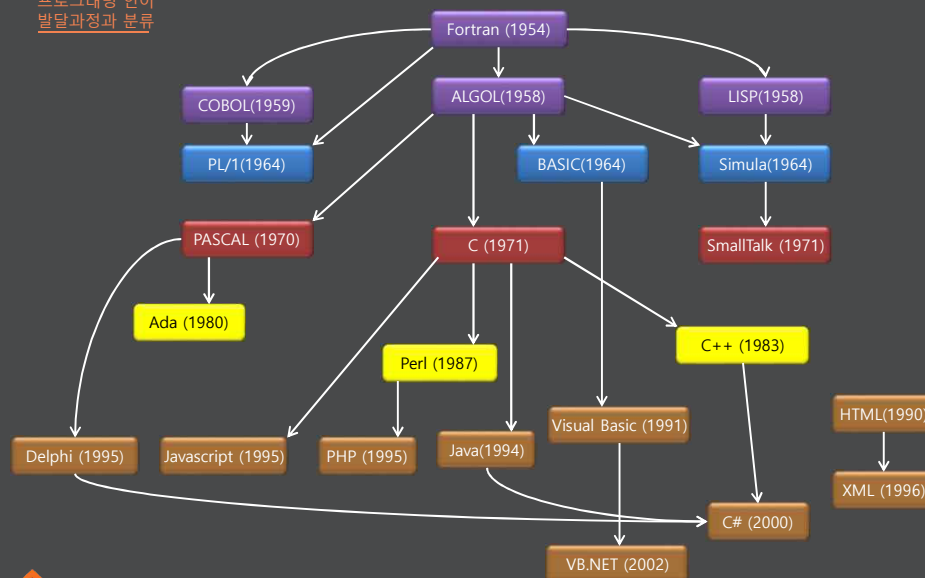


25

## 01

프로그래밍 언어  
발달과정과 분류

## 고급 프로그래밍 언어의 초기와 발달과정



26

## 02

프로그래밍 언어  
발달과정과 분류

## 프로그래밍 언어의 계층별 분류

구분	특징	
저급언어	기계어	2진수를 사용하는 기계 중심의 언어
	어셈블리어	기계어와 1:1로 대응하는 기호로 이루어진 언어 기계어 보다는 쉽다.
고급언어	컴파일러 언어	고급언어로 작성된 프로그램으로, 고급 명령어를 기계어로 번역 FORTRAN, COBOL, C
	인터프리터 언어	프로그래밍 언어의 소스 코드를 바로 실행하는 컴퓨터 프로그램 또는 환경 원시코드를 한 줄씩 실행 BASIC, LISP



27

## 02

프로그래밍 언어  
발달과정과 분류

## FORTRAN (Formular Translator)

1954년 IBM컴퓨터에서 과학 계산을 위해 개발된 고급 언어

수학 연산식의 기술이 용이

Fortran 77 (~1990년대), Fortran 90/95, Fortran 2003, Fortran 2008

최신 버전에서는 VS2008에서 Visual Fortran 실행

Source.f90

```

PROGRAM power
READ (*,*) x,y
Z=x**y
WRITE (*,*) z
END PROGRAM

```



```

17 1.5
70.09280

```



28

## 02

프로그래밍 언어  
발달과정과 분류

## COBOL (COmmon Business-Oriented Language)

제3세대 프로그래밍 언어이고, 1959년 일반 사무처리 언어로 개발되어 지금도 사용

코볼 2002(COBOL 2002)부터 객체 지향 프로그래밍도 포함

Program	PROG 1	Revised by		Page	2	of	5
Programmer	W. O. BERT L. I. B. S.	Date	11-15-1988	Version			
Sequence							
Page	1	of	5				
Line	01	WORKING-STORAGE SECTION					
Line	02	01 DATA-REMAINS-SWITCH	PIC X(02) VALUE SPACES				
Line	03						
Line	04	01 HEADING-LINE					
Line	05	05 FILLER	PIC X(10) VALUE SPACES				
Line	06	05 FILLER	PIC X(12) VALUE 'STUDENT NAME'				
Line	07	05 FILLER	PIC X(10) VALUE SPACES				
Line	08						
Line	09	01 DETAIL-LINE					
Line	10	05 FILLER	PIC X(08) VALUE SPACES				
Line	11	05 PRINT-NAME	PIC X(25)				
Line	12	05 FILLER	PIC X(10) VALUE SPACES				
Line	13						
Line	14	PROCEDURE DIVISION					
Line	15	PREPARE-SENIOR-REPORT					
Line	16	OPEN INPUT STUDENT-FILE					
Line	17	OUTPUT PRINT-FILE					
Line	18	READ STUDENT-FILE					
Line	19	AT END MOVE 'NO' TO DATA-REMAINS-SWITCH					
Line	20						
Line	21	END-READ					
Line	22	PERFORM WRITE-HEADING-LINE					
Line	23	PERFORM PROCESS-RECORDS					
Line	24	UNTIL DATA-REMAINS-SWITCH = 'NO'					
Line	25	CLOSE STUDENT-FILE					
Line	26	PRINT-FILE					
Line	27	STOP-RUN					



29

## 02

프로그래밍 언어  
발달과정과 분류

## BASIC (Beginner's All-purpose Symbolic Instruction Code)

1964년에 교육용으로 개발된 절차형 언어

인터프리터 방식 또는 컴파일러 방식

```
10 PRINT "Hello, world!"
20 END
```

```
INPUT "이름을 입력하세요"; UserName$
PRINT "안녕하세요 "; UserName$
DO
  INPUT "별이 몇 개 필요하세요? "; NumStars
  Stars$ = ""
  Stars$ = REPEAT$(" ", NumStars) '<-ANSI BASIC
  PRINT Stars$
  DO
    INPUT "별이 더 필요하세요(Y/N)?"; Answer$
    LOOP UNTIL Answer$ <> ""
  LOOP WHILE UCASE$(LEFT$(Answer$, 1)) = "Y"
  PRINT "안녕히 가세요 ";
  FOR A = 1 TO 200
    PRINT UserName$, " ";
  NEXT A
  PRINT
```



30

## 02

프로그래밍 언어  
발달과정과 분류

## C (Beginner's All-purpose Symbolic Instruction Code)

1972년에 유닉스 운영체제에서 사용하기 위해 개발된 프로그래밍 언어

시스템 프로그램 개발에 적합 (하드웨어에 대한 직접 제어가 가능)

```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
    return 0;
}
```



31

## 03

프로그래밍 언어  
발달과정과 분류

## 세대별 분류

세대	종류		
1세대	1945년	기계어	
2세대	1950년대 중	어셈블리어	
3세대	1960년대 초	고급 언어. Fortran, COBOL, BASIC, C	절차적 언어
4세대	1970년대 초	초고급 언어, C++, VB, Delphi	문제 해결형 언어(비절차적 언어) (예)마우스나 키보드 이벤트가 일어나면 절차에 상관없이 비절차적으로 실행
5세대	1980년대 초	자연 언어	시각적 그래픽 인터페이스를 통해 3G/4G 언어 컴파일러로 컴파일 할 수 있는 원시코드를 작성

C++, VC++, Boland C++, VB, Java, JavaScript,  
ASP, PHP, JSP, C#, Eclipse



32



## 03

프로그래밍 언어  
발달과정과 분류

## 패러다임에 따른 분류

패러다임 (Paradigm) : 한 시대의 사람들의 견해나  
사고를 근본적으로 규정하고 있는 인식의 체계네트워크가 중심이 되는 시대를 맞아 자바처럼 분산처리가 가능하며, 웹을 기반으로  
하는 프로그래밍 언어 시대가 도래함

분류	특징
객체지향형 (OOP)	<ul style="list-style-type: none"> <li>구조적 프로그래밍의 문제 해결</li> <li>객체 - 기능이 유사한 것끼리 모아놓은 집합체로 속성(property)과 메소드(method)를 포함</li> <li>C++, Java, VB, C#</li> </ul>
함수형	<ul style="list-style-type: none"> <li>함수 호출에 기반한 프로그래밍 접근</li> <li>계산적 측면에서 효율적</li> </ul>
선언형	<ul style="list-style-type: none"> <li>목표를 명시 (전통적인 명령형은 알고리즘을 명시)</li> <li>웹페이지(선언형)는 제목, 글꼴, 그림과 같이 '무엇'이 나타나야 하는지를 묘사하고, '어떤 방법으로' 화면에 출력하는지를 묘사하지는 않음</li> </ul>



33

## 02

프로그래밍 언어  
발달과정과 분류

## C++ 그리고 Visual C++



비야네 스트루스트룹

## C++

- C 언어의 확장 (C 언어 + 객체지향성)이므로 시스템 프로그래밍에 적합
- Class, 연산자 중복, 가상 함수와 상속성

## Visual C++, Visual.NET

- MS사가 개발한 C++언어와 windows의 개발통합환경을 결합하여 Windows 시스템을 쉽게 개발할 수 있는 소프트웨어 언어 및 도구

```
# include <iostream> // std::cout을 제공
using namespace std; // std namespace를 사용할 수 있게 함
int main() {
    cout << "Hello, world!" << endl;
    return 0;
}
```



34

## 02

프로그래밍 언어  
발달과정과 분류

## Visual Basic

- MS사가 Basic과 Windows 통합개발환경을 통합하여 Windows 시스템을 쉽게 개발할 수 있게 만든 소프트웨어 언어 및 도구
- 이벤트 기반 프로그래밍 언어

```
Private Sub Form_Load() ' Hello, World라는 내용의 메시지 상자를 띄웁니다.
    MsgBox "Hello, World!"
End Sub
```



35

## 02

프로그래밍 언어  
발달과정과 분류

## Java



James Gosling

- 가전제품의 네트워크 제어를 목적으로 탄생 (SUN Microsystem 社)
- 현재 웹 어플리케이션 개발과 모바일 기기용 소프트웨어 개발에 널리 사용
- 문법적으로 C 언어에 매우 유사
- Java 언어로 작성된 프로그램을 컴파일러는 바이트코드라는 바이너리 형태로 변환  
이를 실행하기 위해서는 JVM(Java Virtual Machine)이 필요  
따라서, 바이트코드는 CPU나 운영체제의 제한이 없음
- 결과적으로, 플랫폼에 독립적인 객체지향 언어가 탄생
- Eclipse, J builder는 자바와 Windows 통합 개발 환경을 결합한 소프트웨어 언어 및 도구
- 애플릿 (Applet) - HTML에 포함되어 웹 브라우저에서 실행되는 자바 프로그램

HelloWorldApp.java

```
public class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Hello World 를 출력
    }
}
```



36

## 02

프로그래밍 언어  
발달과정과 분류

Tim Berners

## HTML (Hyper Text Markup Language)

- 웹 문서를 작성할 수 있도록 한 표준형식 (일종의 서식언어)
- 기능의 한계
  - Flash - 웹 애니메이션
  - VRML - 3D 구현
  - JAVA - 다양한 효과
  - JavaScript - 동적인 효과
  - ASP - 게시판 방명록

```
<html>
<head>
  <title>Hello HTML</title>
</head>
<body>
  <p>Hello World!</p>
</body>
</html>
```



37

## 02

프로그래밍 언어  
발달과정과 분류

brendan eich

## JavaScript

- 상호작용하는 동적 웹 문서를 작성할 수 있도록 개발된 웹용 스크립트 언어
- JAVA와 직접적인 연관성은 없다.
- 최신 버전 : JavaScript 1.8.5 (2013년 1월 기준)

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html" charset="UTF-8" />
  <title>자바스크립트 페이지</title>
</head>
<body>
  <script type="text/javascript">document.write("<p>Hello World!</p>"); </script>
  <noscript> <p>자바 스크립트 테스트</p> </noscript>
</body>
</html>
```



38

## 02

프로그래밍 언어  
발달과정과 분류

## ASP (Active Server Page)

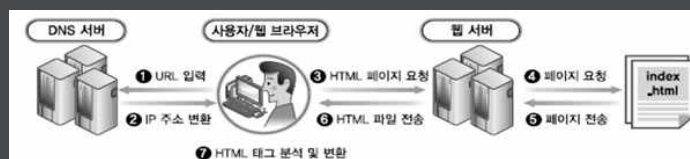
- MS사가 동적인 웹 페이지 생성을 위해 개발한 서버 측 스크립트 엔진
- ASP → ASP.NET
- Client의 요구를 받아 Web Server에서 처리하여 그 결과를 HTML document 형태로 생성하여 Client에게 회송
- 다른 웹 서버용 프로그래밍 언어 : PHP, JSP

```
<html>
<body>
  <% Response.Write "Hello World!" %>
</body>
</html>
```

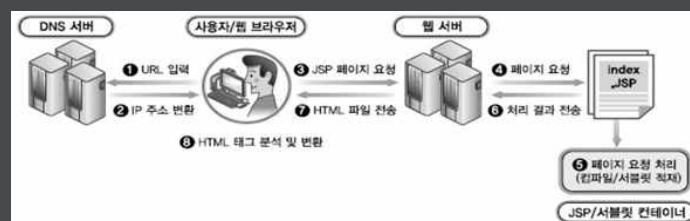
## 02

## 웹 페이지의 동작

## 정적 페이지



## 동적 페이지



## 02

## JSP (Java Server Pages)

Sun Microsystem에서 만든 웹 언어

HTML 내에 JAVA 코드를 삽입하여 웹 서버에서 동적으로 웹 페이지를 생성하여 웹 브라우저로 돌려주는 언어

웹 어플리케이션 서버에서 동작

## 특징

- JAVA의 장점을 사용
- 다양한 servlet 간 데이터 공유
- 많은 사용자의 원활한 접속처리



41

## 03

프로그래밍 언어  
발달과정과 분류

## 인기 있는 프로그래밍 언어와 선택 기준

프로그래밍 언어는 컴퓨터를 제어하기 위해서 사용된다.

프로그래밍 언어도 자연어와 같이 의미를 정의하기 위해서 구문적, 의미적인 규칙을 사용한다.

현재 다양한 프로그래밍 언어 들이 존재하고 , 매년 새로운 것들이 생겨나고 있으나, 대부분의 언어는 많은 사람들이 사용하는 일반적인 언어로 발전하지 못하며, 많은 전문 프로그래머들은 몇 개의 다른 언어를 사용한다.

최근에 인기 있는 프로그래밍 언어에 대한 조사 사용자와 개발자, 산업 현장의 사용 측면 등 바라보는 관점과 조사 방법, 조사 시점에서 차이에 따라 어떤 프로그래밍 언어가 가장 인기 있는지 널리 사용되는 측면에 대한 비교가 여러 가지 있다.

여기에서는 TIOBE라는 회사에서 내는 인덱스이며, 검색엔진을 이용해 인기를 측정



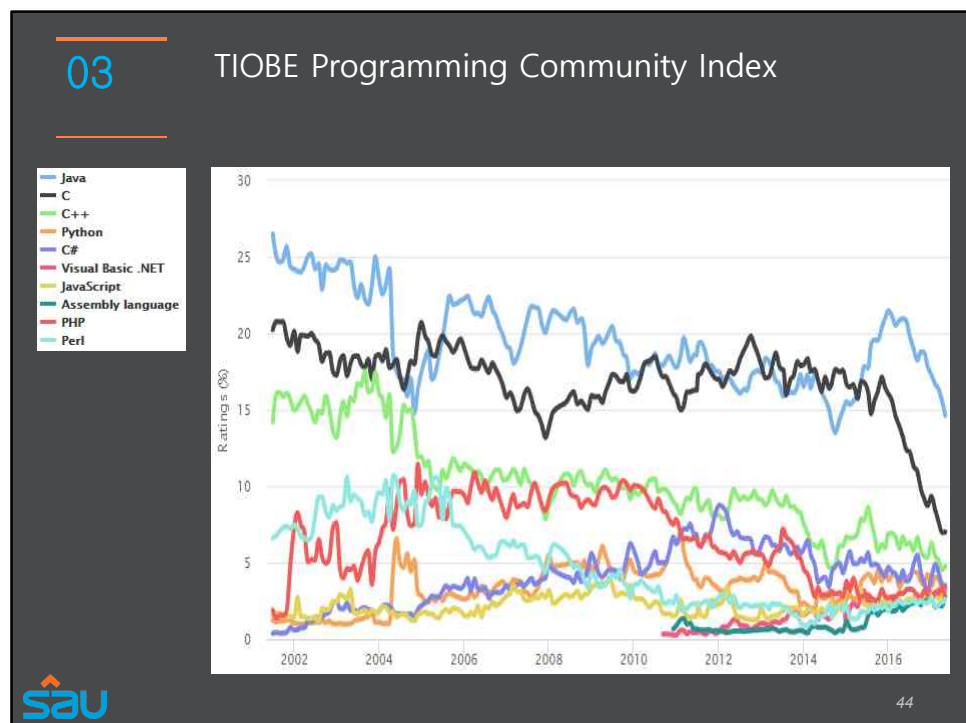
42

03

TIOBE Index for May 2017

	May 2017	May 2016	Change	Programming Language	Ratings
1	1			Java	14.639%
2	2			C	7.002%
3	3			C++	4.751%
4	5		▲	Python	3.548%
5	4		▼	C#	3.457%
6	10		▲▲	Visual Basic .NET	3.391%
7	7			JavaScript	3.071%
8	12		▲▲	Assembly language	2.859%
9	6		▼	PHP	2.693%
10	9		▼	Perl	2.602%


43



## Section 04

## 프로그래밍 코딩의 실제 예

## Chapter 5 프로그래밍 언어

Image from <http://duduchina.co.kr>

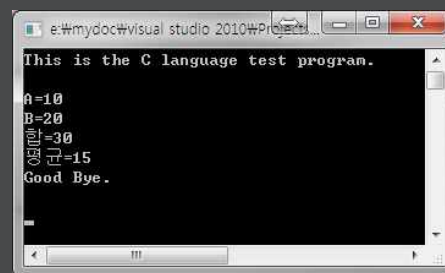
## 01

프로그래밍 코딩의  
실제 예

## 절차적 프로그래밍 언어

## C 언어 프로그래밍 실행 결과

```
#include<stdio.h>
main() {
    int a=10;
    int b=20;
    int sum,avg;
    sum=a+b;
    avg=sum/2;
    printf("This is the C language test program.\n\n");
    printf("A=%d\n",a);
    printf("B=%d\n",b);
    printf("합=%d\n",sum);
    printf("평균=%d\n",avg);
    printf("Good Bye.\n\n");
}
```



## 02

## 객체 지향형 프로그래밍 언어

프로그래밍 코딩의  
실제 예

자바 언어 프로그래밍 실행 결과

```

public class Jv_3_2 {
    public static void main(string[] arg) {
        int x=100, y=30;

        System.out.println("***** result *****");
        System.out.println("x+y =" + (x+y));
        System.out.println("x-y =" + (x-y));
        System.out.println("x*y =" + (x*y));
        System.out.println("x/y =" + (x/y));
        System.out.println("x%y =" + (x%y));
        System.out.println("x^3 =" + (y*y*y));
    }
}

```

```

C:\WINDOWS\system32
***** result *****
x+y =130
x-y =70
x*y =3000
x/y =3
x%y =10
y^3 =27000
C:\source>_

```



47

## 02

## 객체 지향형 프로그래밍 언어

프로그래밍 코딩의  
실제 예

자바 어플리케이션의 프로그래밍 과정



48



## 03

프로그래밍 코딩의  
실제 예

## 웹 기반 프로그래밍 언어

```

<html>
<head>
<title> 자바스크립트 </title>
HTML 머리부분 <br>
<script language="JavaScript">
//자바 스크립트 머리 시작
document.write ("JavaScript head part<br><br>")
// 자바스크립트 머리 끝
</script>
</head>

```

```

<body>
HTML body part<br>
<script language="JavaScript">
/* 자바스크립트 body start */
document.write ("JavaScript head part<br><br>")
/* 자바스크립트 body end */
</script>
</body>
</html>

```



## 04

프로그래밍 코딩의  
실제 예

## 비절차적 프로그래밍 언어

Project1\_1.vbp

Option Explicit

```

Private Sub Command1_Click()
Text1.FontSize = 24
Text1.Text = "비베 6.0에 오심을 환영!!!"
End Sub

```

```

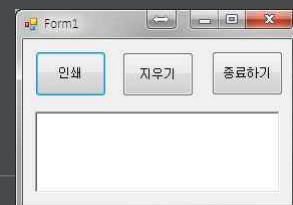
Private Sub Command2_Click()
Text1.Text = " "
End Sub

```

```

Private Sub Command3_Click()
End
End Sub

```



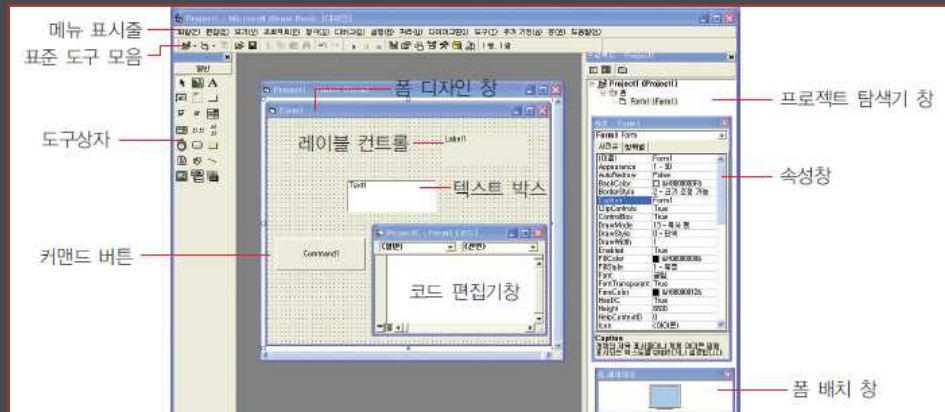
50

## 04

프로그래밍 코딩의  
실제 예

## 비절차적 프로그래밍 언어

비주얼 베이직 6.0의 통합 개발 환경 구성요소



sau

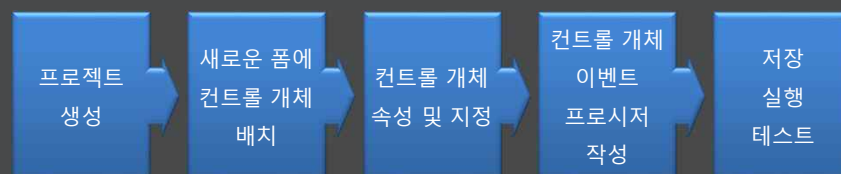
51

## 04

프로그래밍 코딩의  
실제 예

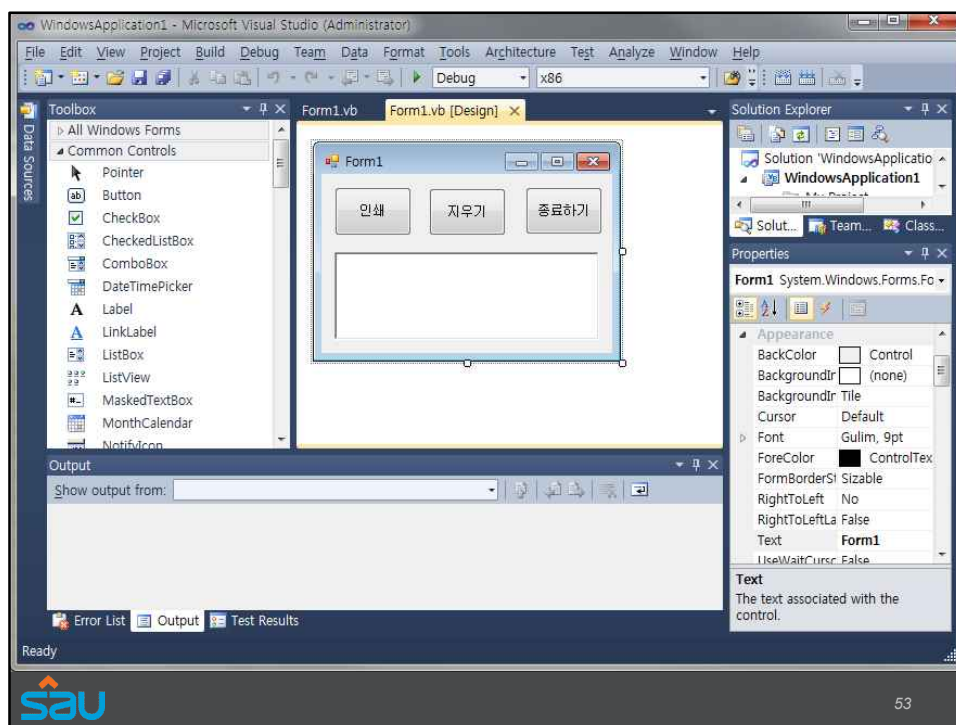
## 비절차적 프로그래밍 언어

비주얼 베이직 6.0의 개략적인 프로그래밍 과정



sau

52



감사합니다