aws re/start

프로그래밍 소개

Python 기본 사항

발표자 이름

날짜

프로그래밍이란?

학습 내용

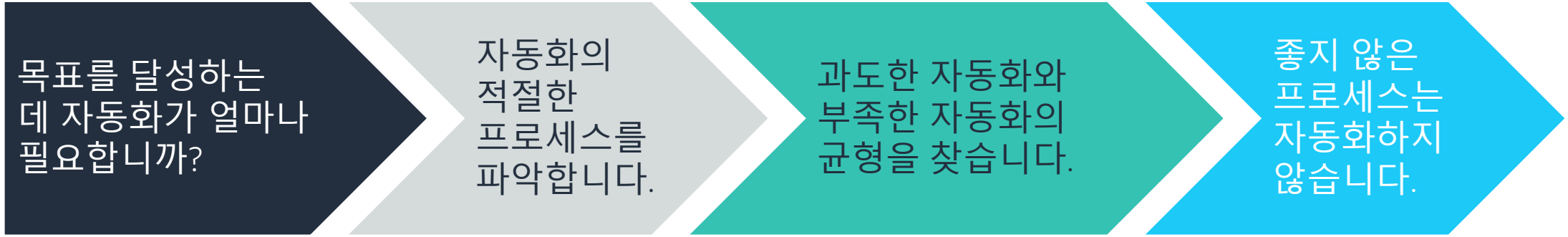
강의 핵심 내용

학습 내용:

- 프로그래밍을 정의합니다.
- 함수와 통합 개발 환경(IDE)의 기능을 설명합니다.
- 프로그래밍 주기를 설명합니다.

프로그래밍의 목적: 자동화

자동화는 시스템, 장비, 프로세스에서 사람의 상호 작용을 제거하는 기술을 일컫습니다.
노동이 많이 요구되는 태스크를 자동화하고 워크플로를 간소화하기 위해 스크립트를 쓰는 경우가 많습니다.
프로세스를 자동화하기 전에 다음 사항을 고려하십시오.



프로그래밍 언어란?

프로그래밍 언어는 지침을 컴퓨터에 전달하는 언어입니다.
지침에는 다음이 포함됩니다.

계산을 수행합니다.

텍스트를 분석합니다.

파일을 읽고 씁니다.

화면에 이미지를 표시합니다.

사용자의 입력을 수락합니다.

소프트웨어 작성 방법

소프트웨어는 텍스트 파일을 사용해 작성합니다.

소프트웨어는 컴퓨터 언어를 사용해 작성합니다.

텍스트 파일을 사용해 소프트웨어 작성

- 일부 텍스트 편집기*에는 프로그래머가 코드를 작성하도록 도와주는 기능이 있습니다.
- 예:
  - Microsoft Visual Studio Code
  - Sublime Text
  - Vi 또는 Vim
  - nano
  - GNU emacs
  - Notepad++
  - TextEdit

소프트웨어는 텍스트 파일을 사용해 작성합니다.

소프트웨어는 컴퓨터 언어를 사용해 작성합니다.

***Microsoft Word는 텍스트 편집기가 아닙니다.**

컴퓨터 언어를 사용해 소프트웨어 작성

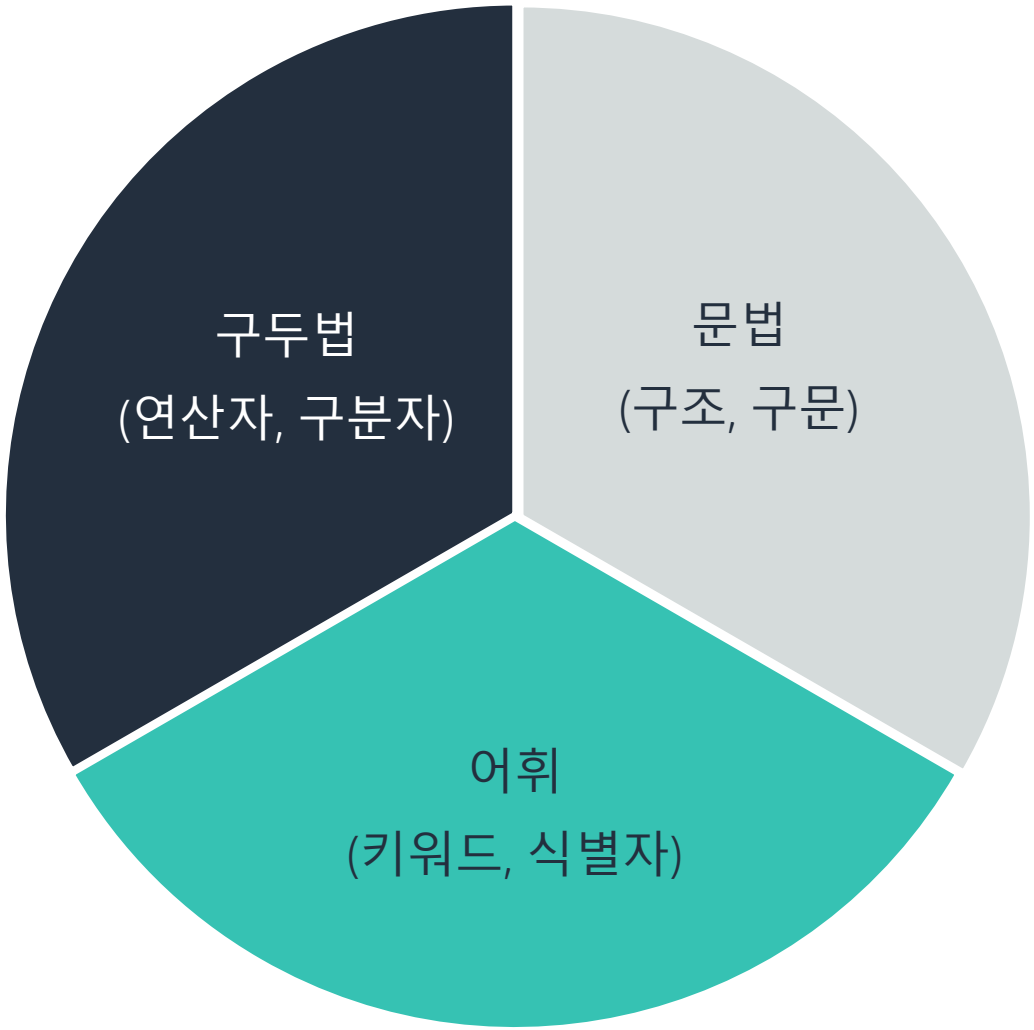
- 다양한 언어가 존재합니다.
- 언어마다 다음과 같은 특징이 있습니다.
  - 문법과 구문
  - 일반적인 용도
  - 커뮤니티
- 예:
  - Python
  - JavaScript
  - C#
  - C/C++

소프트웨어는 텍스트 파일을 사용해 작성합니다.

소프트웨어는 컴퓨터 언어를 사용해 작성합니다.

프로그래밍 언어 요소

인간의 언어와 마찬가지로 프로그래밍 언어에는 다음과 같은 특징이 있습니다.



통합 개발 환경(IDE)

IDE를 통해 단어의 철자가 올바르게 작성되었는지, 어떤 문구가 명확하지 않은지, 잘못 작성한 구문이 있는지 알 수 있습니다.

대부분의 IDE는 문제 해결 방법을 제안합니다.

일부 IDE에서는 한 가지 언어만 사용할 수 있습니다. 예를 들어, PyCharm에서는 Python만 사용할 수 있습니다. PyCharm은 Java 또는 C#의 구문 오류를 표시하지 않습니다.

컴파일러와 인터프리터

컴파일러와 인터프리터는
프로그래머가 개발에 사용하는
고수준의 언어를 머신이 읽을 수
있는 저수준 코드로 변환합니다.

변경 후 코드를 실행하기 전에
컴파일러는 이 변환 과정을
모두 한 번에 수행합니다.

인터프리터는 코드가
실행되는 동안 이 변환 과정을
한 번에 한 단계씩 수행합니다.

컴파일된 언어와 해석된 언어

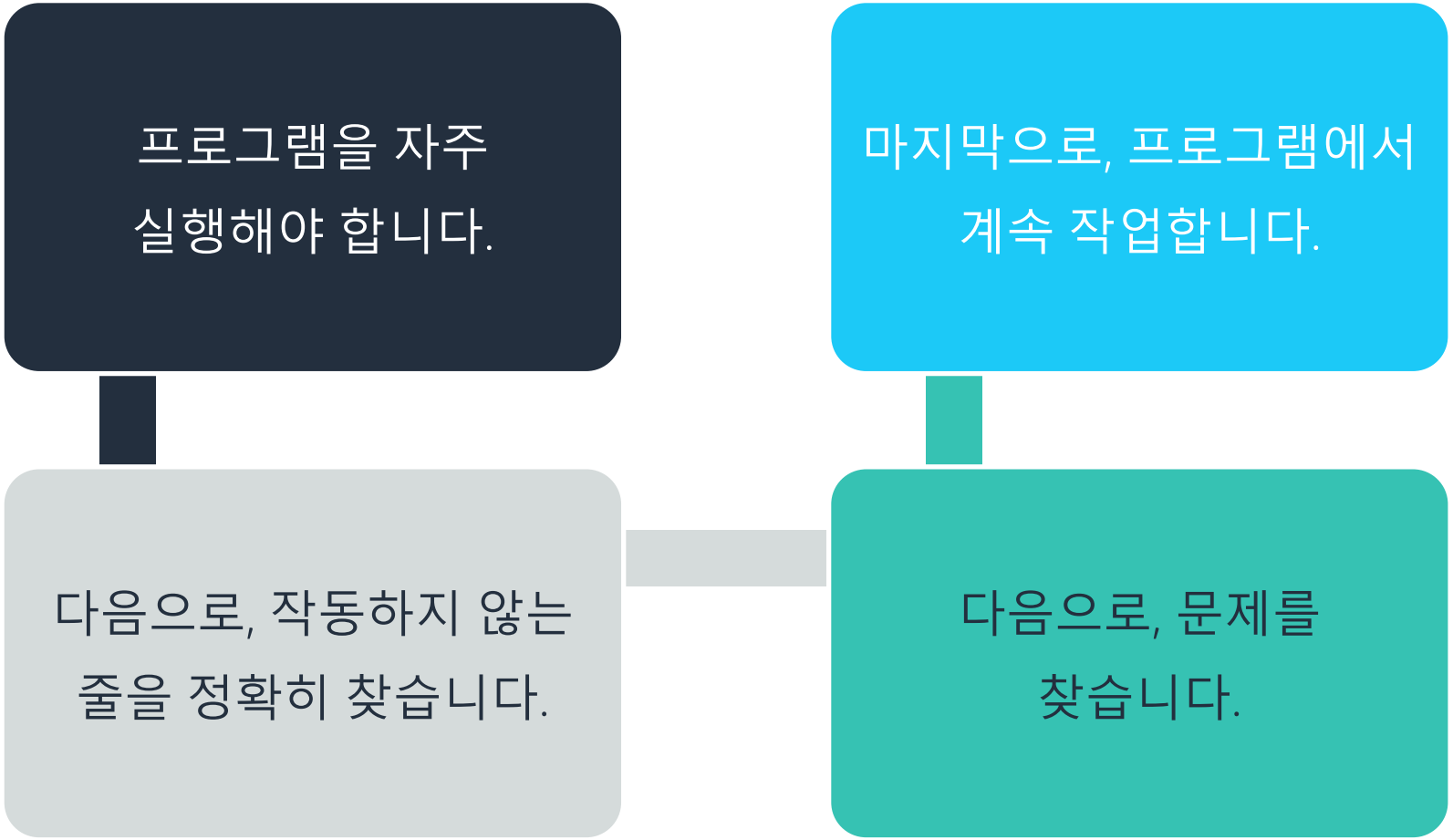
- C/C++
- Basic
- GoLang(Google에서 개발한 언어)

컴파일된 언어

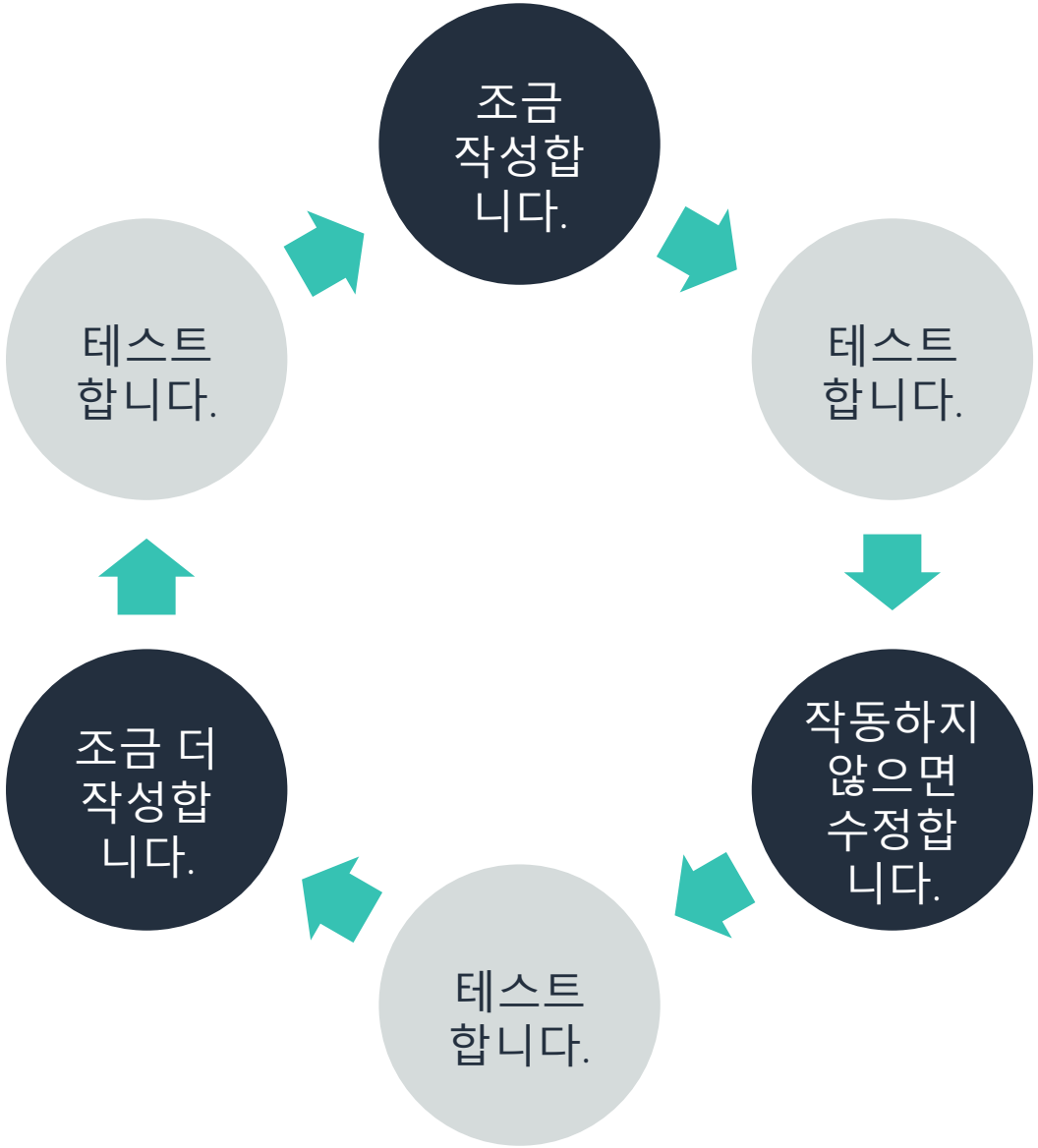
- Python
- Ruby
- JavaScript

해석된 언어

소프트웨어 실행, 디버깅, 수정



소프트웨어가 반복적으로 작성됨



데이터 유형으로 값 범주화

학습 내용

강의 핵심 내용

학습 내용:

- **데이터 유형**이라는 용어를 정의합니다.
- 데이터 유형을 데이터에 올바르게 연결합니다.
- 변수에 값을 할당하고 변수를 데이터 유형에 할당합니다.

데이터 유형이란?

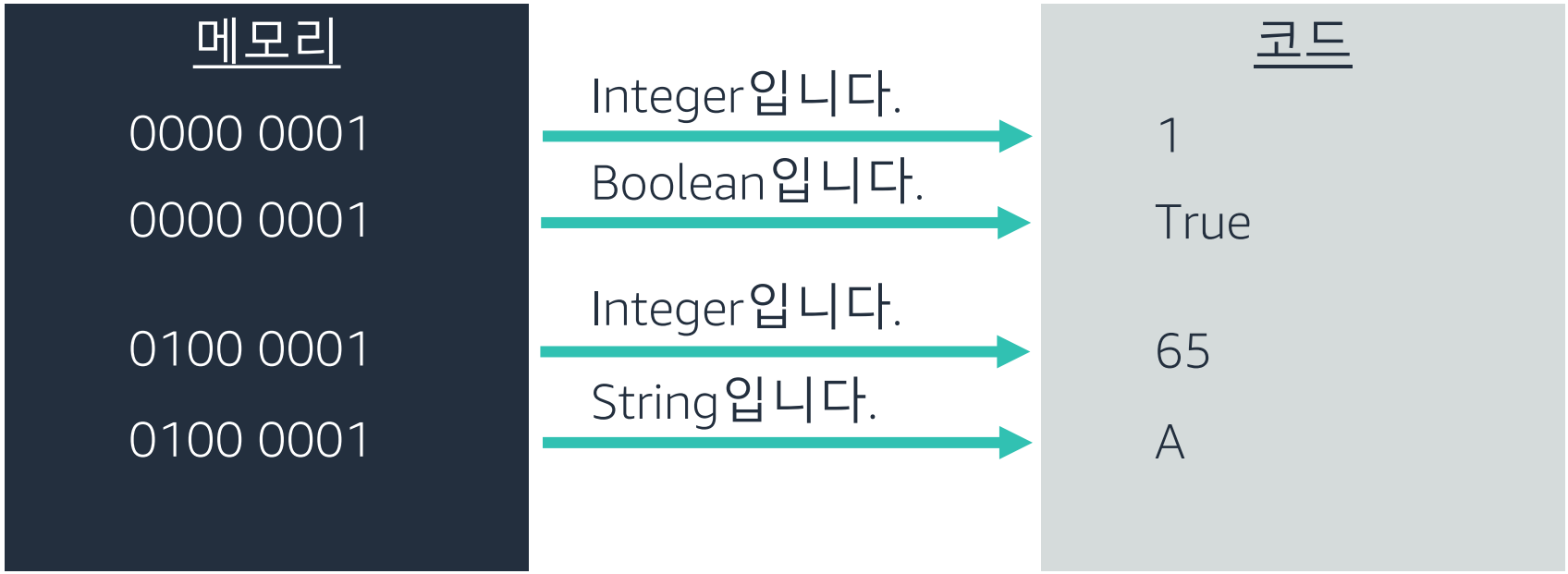
데이터 유형은 값의 분류로, 컴퓨터에 데이터가 어떻게 해석되어야 하는지 프로그래머의 의도를 알려 줍니다.

예:

데이터 값	데이터 유형
45	Integer
290578L	Long
1.02	Float
True	Boolean
“My dog is on the bed.”	String
“45”	String

데이터 유형을 태그해야 하는 이유

- 메모리에서는 모든 것이 0과 1로 구성됩니다.
- 데이터 유형을 지정하면 컴퓨터가 다음을 알 수 있습니다.
  - 데이터 값을 메모리에 인코딩하는 방법
  - 데이터 값을 메모리에서 디코딩하는 방법



aws re/start

활동: 데이터 유형 파악

지침

1. 값마다 데이터 유형을 파악합니다.

    1. “The Martian”
    2. 1.618
    3. 10082L
    4. False
    5. “True”

aws re/start

변수란?

- 변수란 메모리에서 값을 나타내는 코드의 식별자입니다.
- 변수 이름을 사용하면 사람들이 값의 의미를 기억하는 데 도움이 됩니다.

변수에 값 할당

- 거의 모든 언어에는 **할당 연산자**가 있습니다.
- 대부분의 언어는 등호(=)를 사용합니다.

Python
isCoder = True

VB.NET
daysOnJob = 1

F#
daysOnJob <- 1

Pascal
isCoder := true

변수에 데이터 유형 할당

- 일부 언어는 처음 변수를 사용할 때 데이터 유형이 포함되어 있을 것으로 **예상**합니다.

Objective C
int daysOnJob = 1

Java
boolean isCoder = true

변수에 데이터 유형 할당(계속)

- 일부 언어는 할당된 값에 따라 데이터 유형을 **추론**합니다.

Python
count = 10

Swift
var daysOnJob = 1

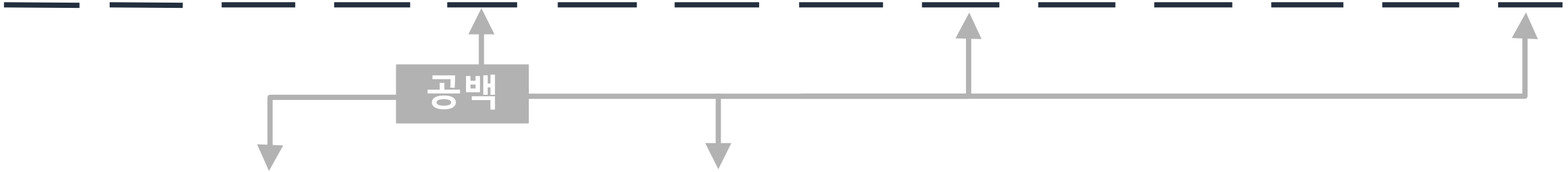
TypeScript
let isCoder = true

활동: 숫자로 문자 표현

American Standards Association for Information Interchange(ASCII)는 인코딩 문자를 컴퓨터에 연결하는 시스템입니다.

ASCII Table에 있는 ASCII 테이블을 사용하여 다음 메시지를 해독하십시오.

72 – 97 – 118 – 101 – 32 – 121 – 111 – 117 – 32 – 115 – 101 – 101 – 110 – 32



109 – 121 – 32 – 99 – 97 – 114 – 32 – 107 – 101 – 121 – 115 – 63

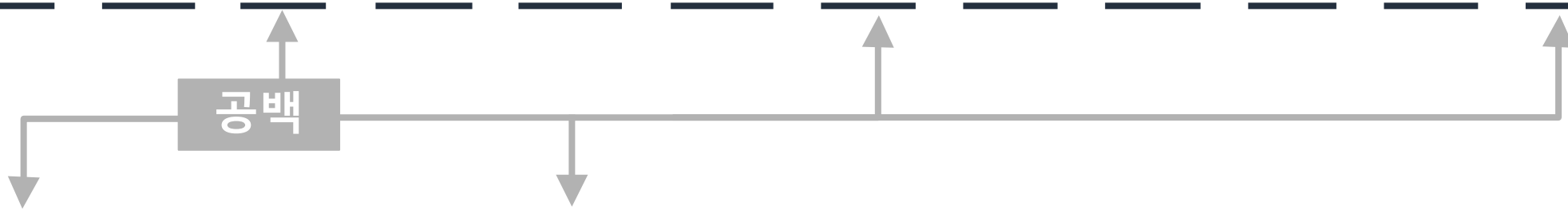
활동: 숫자로 문자 표현

American Standards Association for Information Interchange(ASCII)는 인코딩 문자를 컴퓨터에 연결하는 시스템입니다.

ASCII Table에 있는 ASCII 테이블을 사용하여 다음 메시지를 해독하십시오.

72 – 97 – 118 – 101 – 32 – 121 – 111 – 117 – 32 – 115 – 101 – 101 – 110 – 32

H a v e y o u s e e n



109 – 121 – 32 – 99 – 97 – 114 – 32 – 107 – 101 – 121 – 115 – 63

m y c a r k e y s ?

값을 복합 데이터 유형과 결합

학습 내용

강의 핵심 내용

학습 내용:

- 복합 데이터 유형을 정의합니다.
- 특성 목록에서 복합 데이터 유형을 찾습니다.
- 함수의 속성을 나열합니다.
- 다양한 유형의 컬렉션을 정의합니다.
- 값을 복합 데이터 유형과 결합합니다.

복합 데이터 유형이란?

지금까지는 원시 데이터 유형을 사용했습니다.

원시 데이터 유형

- 수정하지 않고 코딩 언어에 포함된 데이터 유형입니다.

복합 데이터 유형

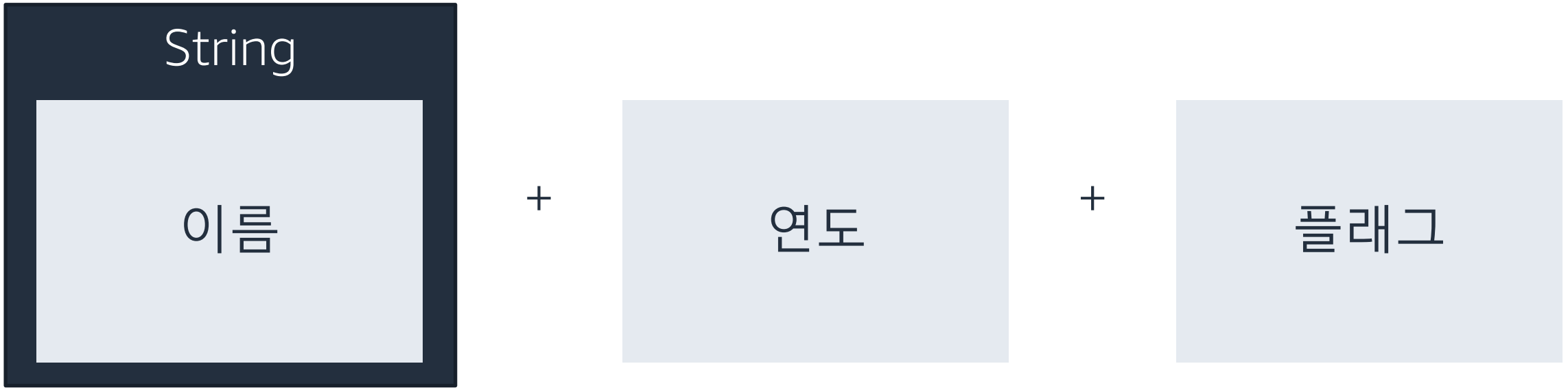
- 여러 데이터 유형을 하나의 단위에 결합합니다.

현대의 모든 프로그래밍 언어에는 복합 데이터 유형을 만드는 방법이 있습니다.

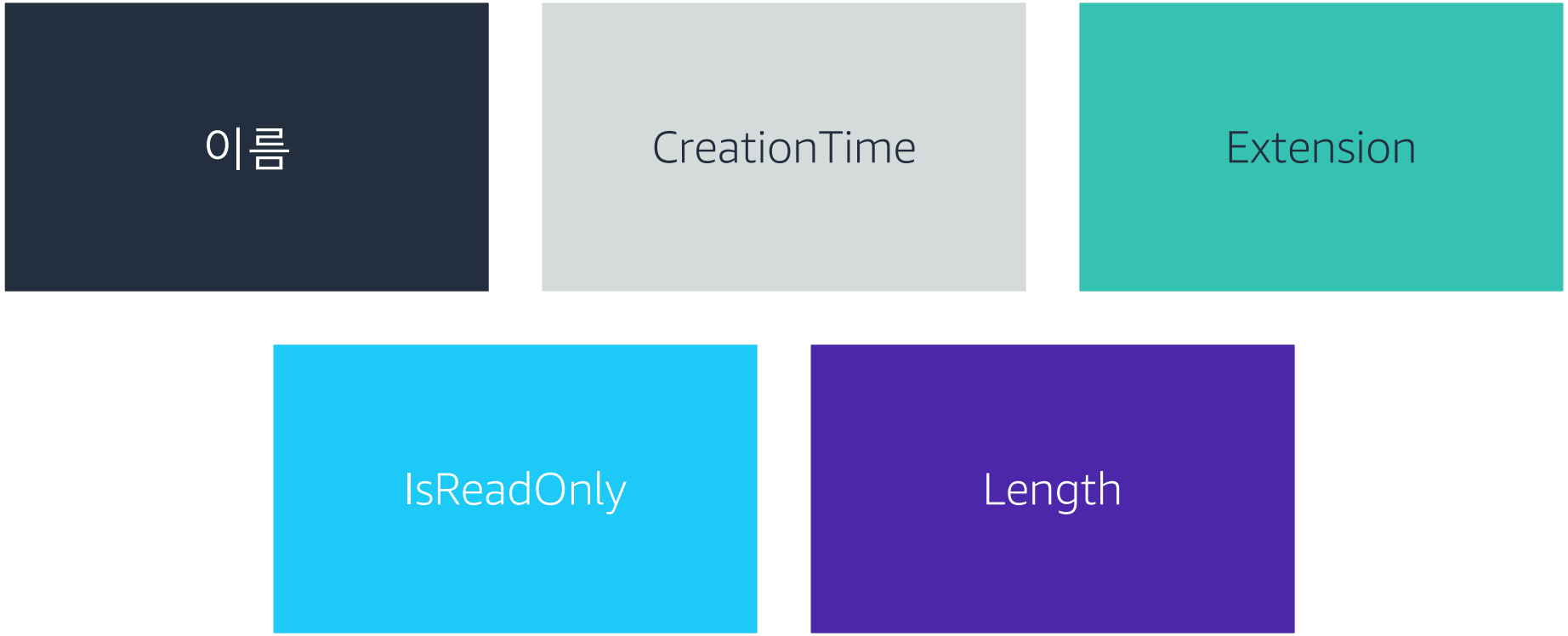
복합 데이터 유형의 예제: 영화

각 영화를 다음으로 설명할 수 있습니다.

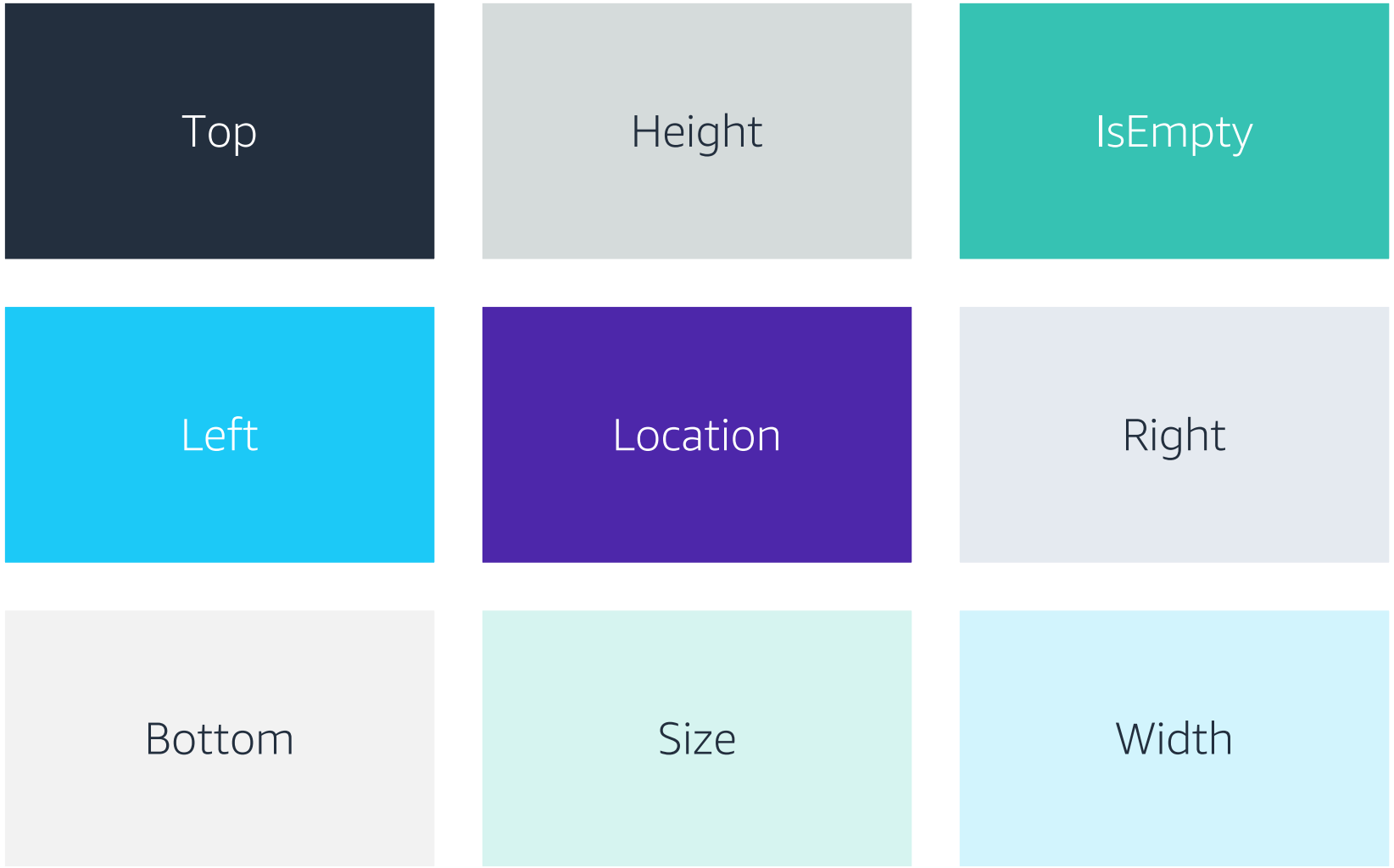
- **이름**(데이터 유형: String)
- 릴리스 **연도**(데이터 유형: Number 또는 Integer)
- 영화의 시청 여부를 표시하는 **플래그**(데이터 유형: Boolean)



복합 데이터 유형 1



복합 데이터 유형 2



복합 데이터 유형 3

연령

문화

설명

성별

이름

Supported_AudioFormats

함수

함수

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

- 다음 슬라이드에는 각 속성의 예가 나와 있습니다.

함수(계속)

함수는 유용한 역할을 합니다.

```
clearScreen()
```

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

```
pi = calculatePi()
```

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

```
area = calculateArea(pi, 4)
```

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

```
showValue(area)
```

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

```
fly(lat, lon, spd, hdg,
vspd, wspd, wdir)
```

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

```
fly(aircraft)

latitude
longitude
speed
heading
verticalSpeed
windSpeed
windDirection
```

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

```
aircraftFuture =
predict(aircraftNow, 60)
```

```
latitude
longitude
speed
heading
verticalSpeed
windSpeed
windDirection
```

함수(계속)

함수는 유용한 역할을 합니다.

함수는 값(변수에 저장할 값)을 반환할 수 있습니다.

함수는 입력 값에 따라 값을 반환할 수 있습니다.

함수는 값을 입력으로 받을 수 있습니다.

함수는 많은 값을 입력으로 받을 수 있습니다.

또는 개발자가 복합 데이터 유형을 만들 수 있습니다.

복합 데이터 유형은 반환될 수 있습니다.

복합 데이터 유형은 어레이일 수 있습니다.

```
collisions =
predict(allAircraft)

latitude
longitude
speed
heading
verticalSpeed
windSpeed
windDirection
```

컬렉션

컬렉션은 여러 값을 하나의 변수로 그룹화합니다. 다양한 유형의 컬렉션을 사용할 수 있습니다.

다음 슬라이드에는 컬렉션의 예가 나와 있습니다.

컬렉션 예제: 어레이

이 테이블은 연령의 기본적인 **어레이**를 보여줍니다. 연령마다 슬롯이 있고 데이터 유형이 같으며 메모리의 이전 연령 및 다음 연령과 인접해 있습니다.

- 첫 번째 슬롯은 모든 언어에서 대부분 0입니다.
- 값은 순서대로 넣지 않아도 됩니다.

슬롯	데이터 유형	값
0	Integer	87
1	Integer	10
2	Integer	2
3	Integer	46
4	Integer	22
5	Integer	19
6	Integer	66

Python 컬렉션 예제: 목록

Python **목록**은 **순서가 있는** 항목의 모음입니다.

- 항목마다 데이터 유형이 다를 수 있습니다.
- 항목은 **인덱스**를 사용하여 액세스할 수 있습니다.
- 항목에는 값이 중복될 수 있습니다.

인덱스	데이터 유형	값
0	String	“John”
1	Integer	55
2	Boolean	True
3	String	“male”
4	String	“Carlos”
5	Integer	22
6	String	“male”

프로그램의 실행 경로 따라가기

학습 내용

강의 핵심 내용

학습 내용:

- 실행 경로가 무엇인지 설명합니다.
- 루프, 조건문, 스위치 등 다양한 흐름 제어 메커니즘의 용도를 설명합니다.

실행 경로란 무슨 뜻입니까?

- 프로그램을 실행할 때 수행하는 단계의 순서입니다.
- 프로그램은 다음과 같이 실행될 수 있습니다.
  - 둘 중 하나(either-or) 선택을 만날 수 있습니다.
  - 선다형을 만날 수 있습니다.
  - 루프로 각 항목에서 작업을 수행할 수 있습니다.
- 프로그래머는 다음 상황에서 단계가 어떨지 예측할 수 있어야 합니다.
  - 코드를 처음부터 작성할 경우
  - 발생한 문제를 디버깅할 경우

조건문

- 모든 프로그래밍 언어에는 코드에서 둘 중 하나의 경로를 선택할 방법이 있습니다.

```
if (guess > number):
    print("Too high!")
elif (guess < number):
    print("Too low!")
else
    print("Exactly right!")
```

Python 예제

```
if(name == 'Juan'):
    bonus = 300
else:
    bonus = salary * 0.1
```

스위치

- 대부분의 프로그래밍 언어에는 편리하게 한 값에서 발생 가능한 여러 상황을 처리하는 방법이 있습니다.

```
switch(sign):
    case "Stop":    pressBreak()
    case "Merge":   accelerate()
    case "Exit":    decelerate()
    default:            ignore()
```

루프

- 모든 프로그래밍 언어에는 컬렉션 내의 각 항목에서 무언가를 수행하는 방법이 있습니다.

```
names = {"wei", "nikki", "akua"}

for name in names:
    newName = capitalize(name)
    print newName
```

Python 예제

```
for(employee in employees):
    employee.bonus = employee.salary * 0.1;
```

버전 관리

학습 내용

강의 핵심 내용

학습 내용:

- 버전 관리의 필요성을 설명합니다.
- Git의 기본 사항을 설명합니다.
- Git와 GitHub의 차이점을 설명합니다.

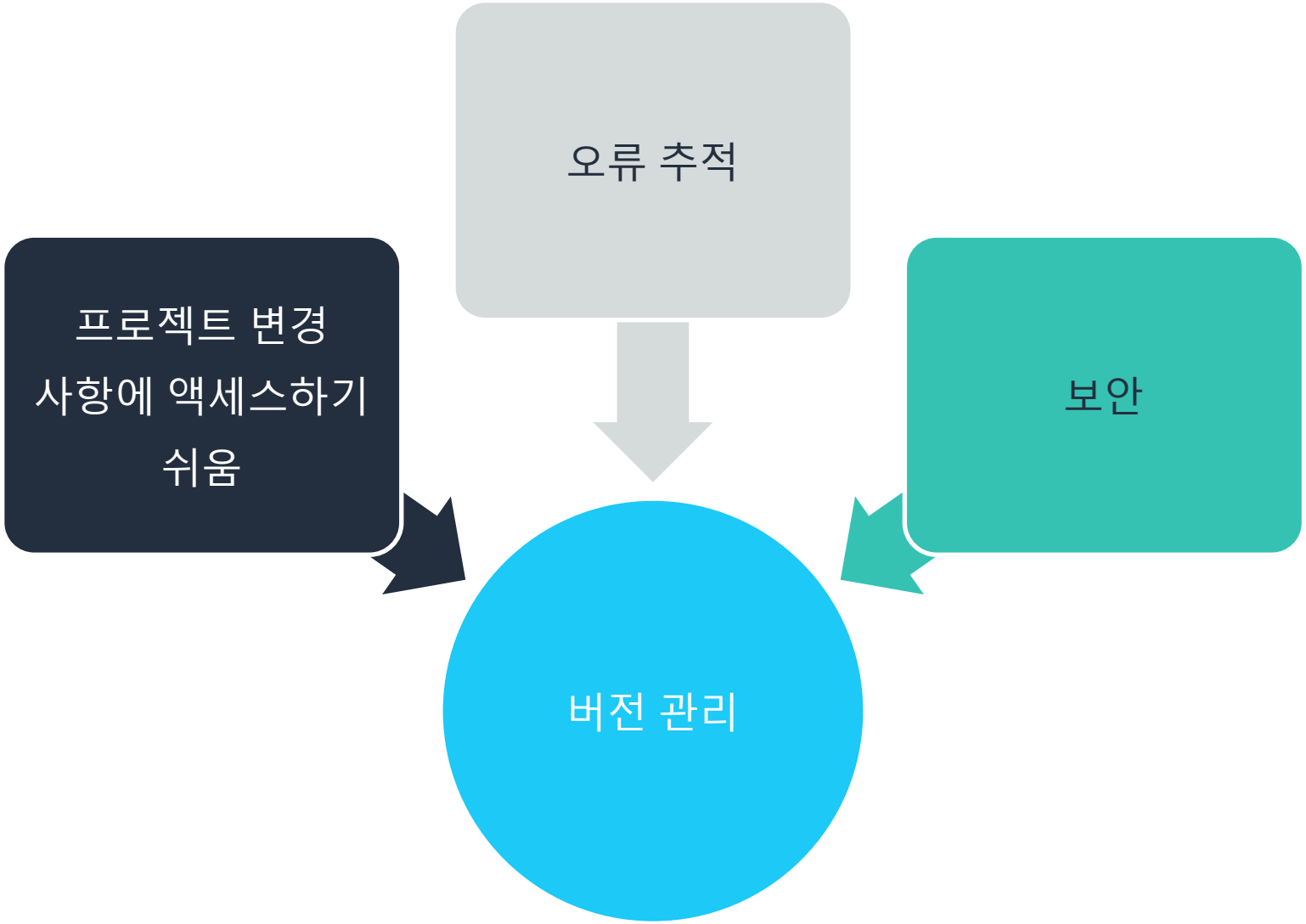
버전 관리와 협업

버전 관리 시스템은 코드의 버전을 추적하고 코드 버전을 업데이트하면 문서화하는 소프트웨어입니다.

버전은 컴퓨터에서 로컬로 수행하거나 버전을 저장하는 전용 웹사이트를 사용해 관리할 수 있습니다.

협업은 여러 사람이 하나의 프로젝트에 작업할 수 있도록 버전 관리를 클라우드 또는 전용 웹사이트에서 수행하는 것을 일컫습니다.

버전 관리의 장점

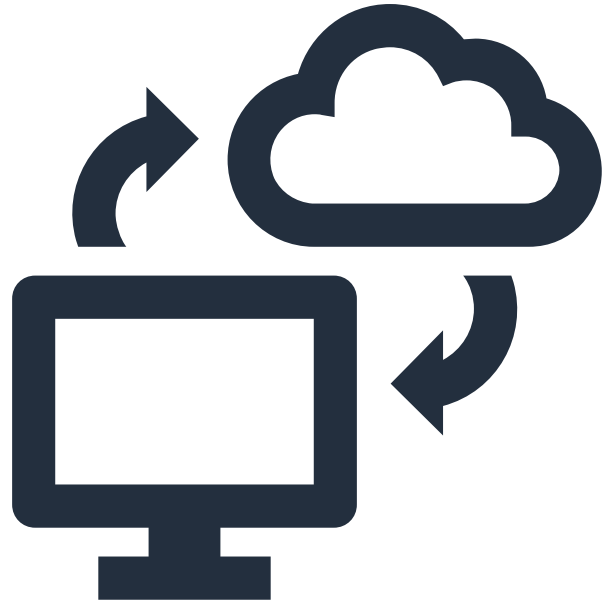


클라우드 인프라 활용

클라우드 또는 전용 웹사이트는 코드의 변경 사항을 저장하는 데 유용합니다.

로컬 컴퓨터에서만 버전을 관리하면 파일의 여러 버전을 저장하는 것보다는 안전하지만 손실되기 쉽습니다.

클라우드에 저장된 데이터는 손실될 위험이 더 적습니다.



버전 관리 도구

다음과 같은 버전 관리 도구를 사용할 수 있습니다.

- Git와 GitHub
- GNU arch
- Mercurial

이 외에도 많은 버전 관리 도구가 있지만, 이 과정에서는 Git와 GitHub에 중점을 둡니다.

Mercurial  GitHub

Git

- Git Tools를 설치합니다.
- 파일을 리포지토리에 추가합니다.
- Git 리포지토리의 파일을 편집합니다.
- 기존 리포지토리를 복제합니다.
- 브랜치를 만들고 병합합니다.
- Git 리포지토리에서 기록을 다시 작성합니다.
- 병합 충돌 문제를 해결합니다.

Git(계속)

Git을 배우는 가장 좋은 방법은 직접 경험하는 것입니다.

1. 웹사이트의 About Git에서 Git Tools를 설치합니다.
2. Git Tools를 설치한 다음 Git Bash 프로그램을 엽니다.
3. Bash 터미널에서 **mkdir GitTest**를 입력하여 **GitTest**라는 이름의 폴더, 즉 디렉터리를 만듭니다.
4. **cd GitTest**를 입력하여 디렉터리를 변경합니다.
5. **git init**를 입력하여 로컬 Git 리포지토리를 만듭니다.
6. **ls -ldF .***을 입력하고 .git 리포지토리를 찾아 이곳에 새로운 리포지토리가 있는지 확인합니다.
7. 다음으로, **touch newFile.txt**를 입력하여 새 파일을 만듭니다.
8. **nano newFile.txt**를 입력하여 새 파일을 엽니다.

Git(계속)

9. 원하는 텍스트를 입력합니다.
10. nano에서 나가려면 CTRL+X를 누릅니다.
11. 터미널에서 **git status**를 입력합니다.
    빨간 글자로 된 newFile.txt가 표시됩니다.
12. 이제 **git add newFile.txt**와 **git status**를 차례로 입력하여 newFile.txt를 로컬 Git 리포지토리에 추가합니다.
    이제 newFile.txt가 초록색으로 표시됩니다. 커밋할 준비가 되었다는 뜻입니다.
13. 프로세스를 마치려면 **git commit -m "Initial commit"**을 입력한 후 **git status**를 입력하여 Git이 새로운 변경 사항을 추적하고 있지 않은지 확인합니다.

Git(계속)

커밋 메시지가 제대로 작동하지 않았다면 터미널에 Please tell me who you are이라는 메시지가 표시됩니다.

14. 이 메시지를 지우려면 터미널에서 다음 명령 두 개를 입력합니다.
    - git config –global user.email “<you@example.com>”
        - <you@example.com>을 내가 실제로 사용하며 이 과정에서 사용하고자 하는 이메일 주소로 대체합니다.
        - 명령의 나머지 부분은 따옴표(“ ”)까지 변경하지 않고 그대로 입력합니다.
    - git config –global user.name “<Your name>”
        - <Your name>에는 이 과정에서 사용하려는 사용자 이름을 입력합니다. 고유한 이름을 사용하십시오.
        - 명령의 나머지 부분은 따옴표까지 변경하지 않고 그대로 입력합니다.

Git(계속)

15. 마쳤으면 git commit -m “Initial commit”을 다시 입력한 후 git status를 입력하여 Git이 새로운 변경 사항을 추적하고 있지 않은지 확인합니다.

GitHub

GitHub는 리포지토리를 호스트하는 서비스입니다. 리포지토리 및 터미널과 같은 명령을 사용합니다.

아직 GitHub 계정이 없다면 실습을 준비하기 위해 GitHub에서 계정을 만드십시오.

GitHub 같은 사이트는 개발자와 미래의 고용주에게 내 코드를 보여 주는 데 유용한 공간입니다.

학습 내용 확인

- 프로그래밍의 중요한 목적은 무엇입니까?
- 프로그래밍 소스 코드를 저장하는 데 어떤 파일 유형을 사용합니까?
- IDE란 무엇입니까?
- 정수의 데이터 유형은 무엇입니까?
- 프로그래밍에서 함수의 목적은 무엇입니까?
- 많이 사용되는 버전 관리 도구 하나를 예로 들어 보십시오.

요점

- 프로그래밍은 프로세스를 자동화하는 방법입니다.
- 프로그래밍은 언어는 컴퓨터에 지침을 전달하는 방법을 명시합니다.
- 소프트웨어는 프로그래밍 언어를 사용하여 텍스트 파일에 작성하며, 실행할 때 언어는 해석되거나 컴파일됩니다.
- 데이터는 인터프리터 또는 컴파일러가 데이터가 String, Integer, Boolean 또는 기타 데이터 유형 중에 무엇인지 알 수 있도록 **유형**이 구분됩니다.
- 복합 데이터 유형은 하나의 변수에 다양한 데이터 유형을 저장합니다.
- 함수는 프로그램 내에서 반복적으로 호출될 수 있는 지침의 컬렉션입니다.
- 버전 관리를 통해 컴퓨터 프로그램, 문서 또는 기타 정보 컬렉션의 변경 사항을 관리합니다.

aws re/start