

구성 요소 및 사용 방법

상단-메뉴

상단 메뉴

{ width=100% }

RobomationLAB 상단 메뉴에는 동글과 로봇을 연결하거나, 파일을 저장하고 불러오는 등 프로그램에서 자주 사용되는 기능들이 모여 있습니다.

아래에서는 각 메뉴의 기능을 순서대로 설명합니다.

로고

{ width=20% }

RobomationLAB 의 로고입니다.

로고를 클릭하면, RobomationLAB 메인 홈페이지로 이동합니다.

블록코딩 에디터가 활성화되어 있는 경우에는, 로고의 문구가 **Block Composer** 로 표시됩니다.

{ width=20% }

파이썬 / 자바스크립트 에디터가 활성화되면, 로고의 문구가 **Script Composer** 로 표시됩니다.

동글 찾기

{ width=10% }

로봇과 통신할 **동글을 검색하고 프로그램에 연결**할 수 있습니다.

프로그램에서 로봇을 제어하기 위해서는, 로봇과 통신할 동글을 먼저 프로그램에 연결해야 합니다.

이러한 과정을 **페어링** 이라고 합니다.

{ width=40% }

동글 찾기 버튼을 누르면 현재 PC 에서 사용 가능한 동글 목록이 표시됩니다.

목록에서 원하는 동글을 선택한 뒤 **연결** 버튼을 클릭하면, 동글이 프로그램에 연결됩니다.

동글 연결상태 확인하기

한 번 프로그램에 연결됐던 동글은, 그 이후에 프로그램을 사용할 때 자동으로 연결됩니다.

{ width=30% }

프로그램에 동글이 정상적으로 연결되어 있을 경우, 아이콘이 **하늘색**으로 변경됩니다.

{ width=30% }

브라우저 탭에 다음 이미지와 같은 아이콘이 있다면, 동글이 연결된 상태임을 확인할 수 있습니다.

⚠ 주의 사항

{ width=40% }

동글이 다른 프로그램 또는 다른 페이지에 이미 연결되어 있는 경우, 프로그램에 동글이 연결되지 않습니다.

이 경우, 동글이 연결되어 있는 프로그램을 찾아 연결을 해제한 뒤, 이 페이지로 돌아와 다시 연결을 시도하세요.

로봇 선택

{ width=10% }

프로그램에서 **사용할 로봇을 선택**하고, 해당 **로봇의 정보와 전용 블록/스크립트 코드를 등록**할 수 있습니다.

프로그램에서 로봇을 제어하기 위해서는, 사용할 로봇의 정보와 블록을 먼저 프로그램에 추가해야 합니다.

{ width=50% }

로봇 선택 버튼을 누르면, 팝업창에 프로그램에서 사용 가능한 로봇 목록이 표시됩니다.

- 햄스터 S - 햄스터 - 뽀로보 - 터틀 - 비글 - 라쿤봇

원하는 로봇을 선택한 뒤 **추가하기** 버튼을 클릭하면, 해당 로봇의 정보와 전용 블록/스크립트 코드가 프로그램에 등록됩니다.

{ width=40% }

{ width=36% }

로봇이 추가되면 다음 항목이 생성됩니다.

- **블록 컴포저 (Block Composer):** 왼쪽 **블록 모음**에 해당 로봇의 전용 **블록** 생성
- **스크립트 컴포저 (Script Composer):** 왼쪽 **코드 모음**에 해당 로봇의 전용 **스크립트 코드** 생성

이를 통해 센서·모터·LED 등 실제 로봇 하드웨어를 자유롭게 움직이고 제어할 수 있습니다.

☒ 참고

RobomationLAB에서는 로봇의 종류, 수에 관계없이 원하는 만큼 로봇을 연결해 사용할 수 있습니다.

단, 여러 대의 로봇을 동시에 연결해 사용하고 싶은 경우,

사용하고 싶은 로봇 수만큼의 동글이 프로그램에 연결되어 있어야 하며, 사용하고 싶은 로봇 수만큼 로봇을 프로그램에 추가해야 합니다.

확장 모듈 선택

{ width=10% }

확장 모듈은 음성 인식, 영상 인식, 이미지 분석 등 AI 기반의 확장 기능을 제공하는 모듈입니다.

프로그램에서 사용할 확장 모듈을 선택하고, 해당 확장 모듈의 정보와 전용 블록/스크립트 코드를 등록할 수 있습니다.

{ width=80% }

확장 모듈 선택 버튼을 누르면 팝업창에 프로그램에서 사용 가능한 확장 모듈 목록이 표시되어 있는 화면이 나타납니다.

- 음성 인식 - 얼굴 찾기 - 상세하게 얼굴 찾기 - 나이, 성별, 표정 - 손 찾기 - 몸 찾기 - 사물 찾기 - 색깔 찾기

원하는 모듈을 클릭하면, **로봇 선택**과 마찬가지로 해당 확장 모듈의 정보와 전용 블록/스크립트 코드가 프로그램에 등록됩니다.

얼굴 찾기, 손 찾기 등 카메라를 사용하는 확장 모듈을 프로그램에 추가하면,

미리보기 - 카메라 탭에 카메라 모듈이 생성되며, 프로그램에 카메라를 연결해 사용할 수 있습니다.

더이상 선택한 확장 모듈이 필요하지 않을 경우, **우클릭 → 제거하기**를 통해 로봇을 목록에서 제외할 수 있습니다.

파일

{ width=10% }

{ width=15% }

코드를 새로 만들거나, 작성한 코드를 파일로 저장하고 불러오는 등 파일을 관리할 수 있습니다.

- 새로 만들기: 현재 활성화된 에디터에서 작성 중인 코드를 초기화하고 새 코드를 생성합니다.
- 저장하기: 현재 작성 중인 코드 파일을 저장합니다. 사용자 컴퓨터의 '다운로드'폴더에 파일이 저장됩니다.
- 다른 이름으로 저장하기: 새로운 이름으로 코드 파일을 저장합니다. 사용자 컴퓨터의 '다운로드'폴더에 파일이 저장됩니다.
- 불러오기: 사용자 컴퓨터에 있는 '.block'형식 (확장자) 의 파일을 불러옵니다.
- 초기화: 프로그램에 등록된 모든 데이터 (로봇 정보, 블록/스크립트 코드, 설정 등) 가 초기화됩니다.

편집

{ width=10% }

{ width=15% }

작업을 취소하거나 다시 실행할 수 있는 기능입니다. - 되돌리기 (Ctrl+Z): 직전 작업을 취소합니다. - 다시하기 (Ctrl+Y): 되돌린 작업을 다시 실행합니다.

소리

{ width=10% }

코딩에 활용할 소리를 선택하거나 직접 오프라인에 있는 소리를 프로그램에 추가할 수 있습니다.

소리 선택하기

{ width=80% }

소리 버튼을 누르면, 프로그램에서 제공하는 다양한 소리를 선택할 수 있는 화면이 나타납니다.

다음과 같은 기능을 사용할 수 있습니다.

- 소리 검색 - ► 소리 미리 듣기 - 소리 목록 (왼쪽 패널) 에 소리 추가

확장 기능

{ width=5% }

왼쪽 하단의 **확장** 버튼 (빨간 박스) 을 클릭하거나 마우스를 올려놓으면, 3 개의 확장 기능 옵션이 나타납니다.

다음과 같은 기능을 사용할 수 있습니다. - 로컬 파일 추가하기: 사용자 컴퓨터에 있는 오디오 파일을 추가 - 소리 녹음하기: 직접 녹음하여 소리 추가 - 무작위 소리 추가하기: 전체 소리 리스트 중 랜덤으로 선택된 소리 추가

코딩에서 소리 사용하기

{ width=40% }

소리 목록 (왼쪽 패널) 에 추가된 소리는 코딩에 활용할 수 있습니다.

{ width=50% }

블록 코딩의 경우, **소리 재생하기** 블록의 드롭다운 메뉴에서 원하는 소리를 선택할 수 있습니다.

{ width=50% }

스크립트 코딩의 경우, **Codes - audio** 카테고리의 **playSound** 함수의 하위 옵션에서 원하는 소리를 선택할 수 있습니다.

코드 실행 시, 선택한 소리가 사용자 컴퓨터의 스피커를 통해 재생됩니다.

예제

{ width=10% }

프로그램에 로봇이 추가되어 있는 경우, 로봇 별로 간단한 예제들을 불러와 체험해볼 수 있습니다.

예제 선택하기

{ width=80% }

예제 버튼을 누르면 위와 같은 **예제 선택하기** 화면이 나타납니다.

카테고리 구분과 **검색** 기능을 통해 원하는 예제를 빠르게 찾을 수 있습니다.

예제 불러오기

1. **예제** 메뉴를 클릭해 **예제 선택** 화면을 열고, 원하는 예제를 선택합니다.
2. 화면이 새로고침되면서 예제가 코딩 영역에 나타납니다.
3. 예제를 불러온 뒤에는 별도의 작업 없이 **실행 버튼 (▶)** 을 눌러 동작을 확인할 수 있습니다.

코드 실행 / 중지

실행 (▶)

{ width=10% }

현재 활성화된 에디터에 작성된 블록 코드 또는 스크립트 코드를 해석해 실행합니다.

작성된 코드에 따라 프로그램에 연결된 로봇을 제어할 수 있습니다.

코드 실행 중에는, 작성되어 있는 코드를 수정할 수 없습니다.

중지 (■)

{ width=10% }

코드 실행을 중지합니다.

에디터 설정

{ width=10% }

{ width=15% }

블록코딩, 파이썬, 자바스크립트 중 원하는 에디터를 선택해서 코딩할 수 있습니다.

에디터를 변경해도 이전에 작성한 코드는 그대로 유지되며, 언제든지 이어서 코딩을 계속할 수 있습니다.

※ 각 에디터들은 서로 독립적으로 동작합니다!

블록코딩 에디터

블록코딩을 선택할 경우, 로고가 **Block Composer**(블록 컴포저) 로 변경됩니다.

블록코딩 에디터에서는 **미리보기 - 코드** 탭을 통해 파이썬, 자바스크립트로 코드가 변환되는 것을 실시간으로 확인할 수 있습니다.

하지만 이 코드들이 **파이썬** 또는 **자바스크립트 에디터**에 자동으로 반영되지는 않습니다.

파이썬 / 자바스크립트 에디터

파이썬 또는 자바스크립트를 선택할 경우, 로고가 **Script Composer**(스크립트 컴포저) 로 변경됩니다.

블록코딩 에디터와 마찬가지로, **파이썬** 또는 **자바스크립트 에디터**에서 작성한 코드는 **블록코딩 에디터**에 자동으로 반영되지 않고 별도로 관리됩니다.

AI 코딩

```
{ width=10% }
```

```
{ width=15% }
```

구글 **Gemini** 의 인공지능 비서 **Gem** 을 활용해 나만의 **AI 코딩 도우미**를 만들고, AI 와 함께 코딩을 할 수 있습니다.

바이브 코딩

바이브 코딩은 복잡한 코딩 지식 없이도 AI 를 활용해 원하는 기능이나 아이디어를 코드로 만들어내는 새로운 개발 방식입니다.

AI 와의 협업을 통해 보다 쉽게 코딩을 학습할 수 있습니다.

```
{ width=40% }
```

코드 리뷰

코드 리뷰에서는 AI 코딩 도우미에게 작성한 코드에 대한 리뷰나 평가를 받을 수 있습니다.

코드 실행 중 에러가 발생한 경우에도, AI 에게 에러가 발생한 이유와 해결 방법을 요청해 빠르게 문제를 해결할 수 있습니다.

```
{ width=40% }
```

AI 코딩 - 시작 가이드

AI 코딩 기능을 처음 사용하는 사용자를 위한 안내 페이지입니다.

AI 코딩 도우미를 생성하고 사용하는 방법, 바이브 코딩과 코드 리뷰를 활용하는 방법을 단계별로 설명합니다.

{ width=50% }

코드 복사

{ width=10% }

현재 활성화된 에디터에 작성된 코드를 클립보드에 복사할 수 있습니다.

Block Composer (블록코딩)

미리보기 - 코드 탭에 활성화되어 있는 언어에 따라 복사 결과가 결정됩니다. - 파이썬 선택 → 파이썬 코드 복사 - 자바스크립트 선택 → 자바스크립트 코드 복사

Script Composer (파이썬 / 자바스크립트)

에디터에 작성되어 있는 코드가 그대로 복사됩니다.

복사한 코드는 Ctrl+V 를 통해 원하는 곳에 붙여넣기 할 수 있습니다.

설정

{ width=10% }

{ width=10% }

프로그램의 기본 설정을 수행할 수 있습니다.

언어

프로그램에 표시되는 언어 (국적) 를 변경합니다.

사용 가능한 언어는 한국어와 영어 (English) 입니다.

동글

- 연결 끊기: 프로그램에 연결된 모든 동글의 연결을 해제합니다.
- 연결 하기: 프로그램에 동글을 다시 연결합니다.

도움말

{ width=10% }

{ width=10% }

프로그램 사용에 필요한 가이드와 외부 자료를 확인할 수 있습니다.

- 시작 가이드: 프로그램을 처음 사용하는 사람들을 위한 시작 가이드를 제공합니다.
 - 코드 설명서: Block Compomser, Script Composer 에서 제공하는 코드 문법이 정리된 설명서를 제공합니다.
 - 로보메이션 랩: RobomationLAB 메인 페이지로 이동합니다.
 - 홈페이지: 로보메이션 회사 공식 홈페이지로 이동합니다.
 - 유튜브: 로보메이션 유튜브 페이지로 이동합니다.
 - 쇼핑몰: 로보메이션 쇼핑몰 페이지로 이동합니다.
 - 정보: 프로그램 버전 및 업데이트 내역, 이용약관, 개인정보처리방침 등을 확인할 수 있습니다.
 - 문의하기: 프로그램 사용 중 궁금한 점이나 버그 등을 문의할 수 있습니다.
-

에디터

에디터

에디터는 **블록** 또는 **스크립트 코드**를 이용해, 로봇을 제어하기 위한 코드를 작성할 수 있는 영역입니다.

아래에서는 **블록코딩 / 스크립트 코딩** (파이썬, 자바스크립트) 환경에서 각각 코딩하는 방법과 주의해야 할 점들을 소개합니다.

블록코딩 에디터

블록 카테고리

{ width=60% }

RobomationLAB 에서 제공하는 블록들을 **카테고리**로 분류한 영역입니다.

카테고리를 클릭하면, 각 카테고리에 해당하는 **블록 모음**을 확인할 수 있습니다.

다음은 기본으로 제공되는 블록 카테고리의 종류입니다.

- 논리
- 반복
- 연산
- 문자열
- 리스트
- 색상
- 소리

- 제어
- 변수
- 함수
- 기타

이외에도, **로봇** 또는 **확장 모듈**을 프로그램에 추가하면, 프로그램에서 전용 블록 모음을 이용할 수 있습니다.

☒ 참고

{ width=24% }

프로그램에 추가한 블록 중 더 이상 사용하지 않는 카테고리는, **마우스 우클릭 → 제거하기**를 통해 블록 카테고리에서 제거할 수 있습니다.

블록 모음

{ width=60% }

각 카테고리의 모든 블록을 모아놓은 영역입니다.

블록 모음에 있는 블록들은 **Drag&Drop** 방식으로 코딩 영역으로 옮길 수 있습니다.

{ width=30% }

☒ 참고

블록의 사용 방법을 확인하고 싶다면, **마우스 우클릭 → 도움말**을 통해 각 블록 별로 사용 방법이 설명되어 있는 도움말을 확인할 수 있습니다.

코딩 영역

{ width=60% }

블록 모음으로부터 가져온 블록을 조립할 수 있는 영역입니다.

조립된 블록들은 **파이썬/자바스크립트 코드**로 실시간으로 변환되며, 코드를 실행하면, 이 코드들을 해석해 로봇을 움직이고 제어할 수 있습니다.

블록 기본 구조

블록코딩 에디터에서 코딩을 할 때는, 다음과 같은 기본 구조를 지켜야 합니다.

{ width=30% }

블록코딩 에디터에서는 **시작하기**와 **무한 반복하기** 함수 블록 안에 있는 코드를 해석해 실행합니다.

따라서, **시작하기**와 **무한 반복하기** 함수 블록 안에 블록을 넣어 코드를 작성해야 합니다.

시작하기

시작하기 함수 블록 안에는 코드 실행 시 초기에 수행할 동작들을 정의합니다.

기다리기 블록을 활용해, 시간 순서대로 동작이 수행되도록 할 수 있습니다.

무한 반복하기

무한 반복하기 함수 블록 안에는 코드가 실행되는 동안 반복해서 수행할 동작들을 정의합니다.

정의한 동작들을 10ms 에 한번씩 반복해서 수행됩니다.

⚠ 주의 사항

무한 반복하기 함수 블록 안에서는 **기다리기** 기능이 포함된 블록을 사용할 수 없습니다.

잘못하면 코드에 오류가 발생할 수 있으니 주의해야 합니다.

☒ 참고

(함수 카테고리를 통해 생성한 커스텀 함수를 제외하고)

시작하기 또는 **무한 반복하기** 함수 블록 밖에 있는 블록들은, 코드 실행 시에 아무 영향을 주지 않습니다.

블록 사용 방법

블록 추가하기

```
{ width=30% } { width=30% } { width=30% }
```

추가하고 싶은 블록을 **블록 모음**에서 **드래그**하여 **에디터**에 **드롭**하면 해당 블록을 추가할 수 있습니다.

블록 복사/붙여넣기

블록을 선택한 뒤 **Ctrl+C** 키를 누르면, 선택한 블록을 **복사**할 수 있습니다.

Ctrl+V 키를 누르면, 마지막으로 복사한 블록을 에디터에 **붙여넣기** 할 수 있습니다.

블록 삭제

에디터에서 블록을 삭제할 수 있는 방법은 총 3 가지가 있습니다.

1. 블록을 선택한 뒤 **Backspace** 키를 누르면, 선택한 블록을 삭제할 수 있습니다.
2. 삭제하고 싶은 블록을 **에디터**에서 **드래그**하여 **블록 카테고리**에 **드롭**하면 해당 블록을 삭제할 수 있습니다.

```
{ width=30% } { width=30% } { width=30% }
```

3. 삭제하고 싶은 블록을 **에디터**에서 **드래그**하여 **휴지통**에 **드롭**하면 해당 블록을 삭제할 수 있습니다.
삭제된 블록은 **휴지통**에서 다시 확인할 수 있습니다.

```
{ width=8% }
```

여러 블록 동시에 선택하기

Shift 키를 누른 상태로 화면을 드래그하거나 블록들을 클릭하면,
여러 블록을 동시에 선택해서 옮기거나 복사, 삭제할 수 있습니다.

추가 옵션

이외에도 블록을 **마우스로 우클릭**하면,
블록 축소/확장, 활성화/비활성화, 도움말 같은 다양한 추가 옵션들을 확인할 수 있습니다.

스크립트 에디터

코드 모음

```
{ width=60% }
```

로봇 코딩에 필요한 **기본 함수**들과, 로봇/확장 모듈 전용 **스크립트 코드**들을 **카테고리**로 분류한 영역입니다.

다음은 기본 함수 (Codes) 에서 제공되는 코드 카테고리의 종류입니다.

- logic (논리)
- loops (반복)
- math (연산)
- text (문자열)
- lists (리스트)
- color (색상)
- audio (소리)
- control (제어)

기본 함수에서 제공되는 코드들은, 블록코딩 에디터의 기본 블록들과 모두 같은 역할을 수행합니다.

참고

코드 모음을 활용해 스크립트 코딩 에디터에서 코딩하는 방법은 **코드 모음 활용 방법**에서 확인하실 수 있습니다.

코드 에디터

```
{ width=60% }
```

로봇을 제어하기 위한 코드를 작성할 수 있는 영역입니다.

에디터 설정 에서 선택한 에디터 (파이썬 / 자바스크립트) 에 따라, 원하는 프로그래밍 언어로 코드를 작성할 수 있습니다.

코드 기본 구조

코드 에디터에서 코딩을 할 때는, 다음과 같은 기본 구조를 지켜야 합니다.

```
{ width=40% } { width=40% }
```

코드 에디터에서는 **setup** 함수와 **loop** 함수 안에 있는 코드를 해석해 실행합니다.

따라서, **setup** 함수와 **loop** 함수 안에 코드를 작성해야 합니다.

setup

setup 함수 안에는 코드 실행 시 초기에 수행할 동작들을 정의합니다.

wait 함수를 활용해, 시간 순서대로 동작이 수행되도록 할 수 있습니다.

loop

loop 함수 안에는 코드가 실행되는 동안 반복해서 수행할 동작들을 정의합니다.

정의한 동작들을 10ms 에 한번씩 반복해서 수행됩니다.

⚠ 주의 사항

loop 함수 블록 안에서는 **wait** 함수와 같이 **async/await** 기능이 필요한 함수를 사용할 수 없습니다.

잘못하면 코드에 오류가 발생할 수 있으니 주의해야 합니다.

코드 모음 활용 방법

아래에서는 간단한 예시와 함께 **코드 모음을 코딩에 활용하는 방법**에 대해 설명합니다.

원하는 코드 찾기

코드 모음에서는 로봇 코딩에 필요한 다양한 함수 및 코드들을 제공합니다.

```
{ width=30% }
```

Codes 카테고리 안의 메뉴들을 확인해보면, 아이콘이 있는 메뉴들을 확인할 수 있습니다.

logic 메뉴를 한 번 클릭하면, 아이콘이 로 바뀌면서 안에 있는 **하위 메뉴**들을 펼쳐서 확인할 수 있습니다.

이렇게 하위 메뉴를 가지고 있는 메뉴를 **‘카테고리’**라고 합니다.

logic 카테고리 안의 **ternary** 처럼 메뉴에 아이콘이 없다면, 메뉴 안에 더이상 **하위 메뉴가 없다**는 것을 의미합니다.

이렇게 하위 메뉴를 가지고 있지 않은 메뉴를 **‘코드’**라고 합니다.

```
{ width=30% }
```

하위 메뉴가 펼쳐져 있는 카테고리를 다시 클릭하면, 아이콘이 다시 로 바뀌면서 하위 메뉴들이 감춰집니다.

위와 같은 방법으로 카테고리를 따라 가면서, 코드 모음에서 원하는 코드를 찾을 수 있습니다.

에디터에 코드 삽입하기

에디터에 코드를 삽입하는 방법은 다음과 같습니다.

{ width=30% }

메뉴 이름 왼쪽에 { width=1% } 아이콘이 있는 메뉴의 경우, 선택할 수 있는 **코드 옵션**이 없는 메뉴를 의미합니다.
하위 메뉴를 가지고 있는 **카테고리** 메뉴가 대부분 이에 해당합니다.

메뉴 이름 왼쪽에 { width=1% } 아이콘이 있는 메뉴의 경우, **코드 옵션**을 선택할 수 있는 메뉴를 의미합니다.
하위 메뉴를 가지고 있지 않은 **코드** 메뉴가 대부분 이에 해당합니다.

에디터에 삽입하고 싶은 코드를 마우스로 **우클릭**하면, 선택 가능한 옵션들을 확인할 수 있습니다.

{ width=30% }

기본 함수 카테고리 안에 있는 코드들은, 다음과 같은 옵션을 가질 수 있습니다. (예. **logic** 카테고리의 **if** 코드) - **코드 삽입**

미리 지정된 함수 또는 코드를 에디터에 삽입합니다.

{ width=30% }

로봇 코드 카테고리 안에 있는 코드들은, 다음과 같은 옵션을 가질 수 있습니다. (예. **햄스터 S** 카테고리의 **wheel.speed.left** 코드)

- 코드 삽입

미리 지정된 함수 또는 코드를 에디터에 삽입합니다.

대부분 로봇마다 제공되는 **커스텀 함수** (functions) 카테고리의 코드를 에디터에 삽입할 때 확인할 수 있습니다.

• 값 가져오기

로봇의 바퀴 속도, LED 색 같이 사용자가 설정할 수 있는 값이나 로봇의 센서 값 등 값을 확인할 수 있는 코드를 에디터에 삽입합니다.

```
# 햄스터 S 의 왼쪽 바퀴 속도 값
__('HamsterS*0:wheel.speed.left').d
```

• 값 쓰기

로봇의 바퀴 속도를 설정하거나, 이동할 거리를 설정하는 등 로봇을 제어하기 위해 사용자가 값을 써야할 때 사용하는 코드를 에디터에 삽입합니다.

```
# 햄스터 S 의 왼쪽 바퀴 속도를 50 으로 설정합니다.
__('HamsterS*0:wheel.speed.left').d = 50
```

• 기다리기

로봇이 이동할 거리를 설정한 후 이동이 완료될 때까지 기다리는 등 시간 순서대로 로봇의 동작을 제어하기 위해 사용하는 코드를 에디터에 삽입합니다.

대부분 이름 앞에! 가 붙은‘Event’타입 코드들에서 확인할 수 있습니다.

```
# 햄스터 S 가 5cm 이동하며, 이동이 완료될 때까지 기다립니다.  
__('HamsterS*0:wheel.move').d = __getDistance('HamsterS', 5, 'cm')  
await __('HamsterS*0:wheel.!move').w()
```

• 이벤트 확인

로봇의 이동이 완료되었는지 또는 로봇에 두드림이 발생했는지 등 특정 이벤트가 발생했는지 확인할 수 있는 코드를 에디터에 삽입합니다.

대부분 이름 앞에! 가 붙은‘Event’타입 코드들에서 확인할 수 있습니다.

```
if __('HamsterS*0:wheel.!move').e == True:  
    print("Event")
```

상수 값을 갖는 코드

```
{ width=30% }
```

코드 메뉴 중에는, 다음과 같이‘하위 메뉴를 가지면서도 코드를 삽입할 수 있는 메뉴’가 존재합니다. (예. **audio** 카테고리의 **playSound** 코드)

이러한 코드를‘상수 값을 갖는 코드’라고 하며,
이 메뉴의 하위 메뉴에 있는 코드들을‘상수 값’이라고 합니다.

이러한 코드들은,
메뉴를 클릭해 **하위 메뉴**를 펼치면 사전에 입력값으로 지정되어 있는 상수 값들을 확인할 수 있습니다.
또한 **마우스 우클릭**을 통해 **코드 옵션**을 선택하여 에디터에 코드를 삽입할 수 있습니다.

☒ 참고

```
{ width=30% }
```

프로그램에 추가한 로봇 / 확장모듈 전용 코드 중 더 이상 사용하지 않는 코드 카테고리는,
마우스 우클릭 → 제거하기를 통해 코드 모음에서 제거할 수 있습니다. —

미리보기

미리보기

미리보기는 **로봇 연결상태**를 확인하거나 코드, 카메라 콘솔 등 로봇 코딩을 보조하는 역할을 하는 영역입니다. 아래에서는 각 탭에서 이용할 수 있는 기능들에 대해 설명합니다.

로봇 연결상태 탭

{ width=40% }

로봇 연결상태 탭에서는 현재 연결되어 있는 로봇들의 정보를 확인할 수 있습니다.

실시간 센서 값 확인하기를 클릭하면, 로봇의 센서 값을 그래프 또는 숫자를 통해 실시간으로 확인할 수 있습니다.

{ width=40% }

1. 조회할 센서를 선택할 수 있으며, 로봇별로 지원되는 센서 종류가 다릅니다.
선택한 센서에 따라 그래프 및 데이터 표시 방식이 해당 센서의 값에 맞춰 자동으로 변경됩니다.
2. 센서 값을 몇 초 간격으로 확인 또는 기록할지 선택할 수 있습니다.
선택한 주기에 따라, 데이터를 기록하고 그래프에 표시할 수 있는 최대 시간이 달라집니다.
3. 센서 그래프 위에 마우스를 올려 놓으면, 해당 시점의 센서 값을 실시간으로 함께 확인할 수 있습니다.
4. 기록 시작/중지: 센서 데이터를 기록합니다.
5. 내보내기: 기록된 센서 데이터 (.csv) 와 그래프 화면 (.png) 을 파일로 저장합니다.
이 기능은 데이터를 기록한 후에만 사용할 수 있습니다.
6. 캡처: 현재 그래프 화면 (.png) 을 실시간으로 저장합니다.
센서 값의 변화를 기록하고, 기록된 데이터 파일과 그래프 화면을 파일로 저장할 수 있습니다.

코드 탭

{ width=40% }

코드 탭에서는 블록을 추가하거나 제거할 때 실시간으로 **파이썬, 자바스크립트** 코드에 반영되는 모습을 확인할 수 있습니다.

블록이 어떻게 코드로 변환되는지 함께 확인하면서, 보다 쉽게 코딩 문법을 학습할 수 있습니다.

※ 코드 탭은 **블록코딩 에디터**에서만 제공됩니다.

파이썬 / 자바스크립트 에디터에서는 사용자가 직접 코드를 작성하기 때문에 별도로 코드 탭이 제공되지 않습니다.

카메라 탭

{ width=40% }

카메라 탭에서는 PC 에 연결된 카메라 화면을 실시간으로 확인할 수 있습니다.

이 기능은 **카메라를 사용하는 확장 모듈을 프로그램에 추가한 경우**에만 활성화되며, 카메라 탭에 카메라 모듈이 생성됩니다.

각 카메라 모듈에서는 다음 기능을 사용할 수 있습니다.

- 현재 PC 에서 사용 가능한 카메라 중 하나를 선택하여 화면을 표시할 수 있습니다.
- **켜기 / 끄기** 버튼을 통해 카메라 화면을 표시하거나 중지할 수 있습니다.

콘솔 탭

콘솔 탭에서는 프로그램 실행 중에 출력되는 로그 메시지 (Log) 와 스코프 (Scope) 그래프를 실시간으로 확인할 수 있습니다.

이 탭은 디버깅, 센서 값 모니터링, 그래프 기반 분석 등에 활용됩니다.

로그 (Log)

로그 출력하기 블록을 사용하면, 지정한 태그와 함께 텍스트 또는 숫자 값을 콘솔 로그 영역에 출력할 수 있습니다.

{ width=30% }

{ width=40% }

스코프 (Scope)

스코프 출력하기 블록을 사용하면, 태그별로 숫자 값을 **실시간 그래프**로 표현할 수 있습니다.

최솟값·최댓값·그래프 색상을 설정하여 원하는 값의 변화를 시각적으로 확인할 수 있습니다.

{ width=40% }

{ width=40% }
