K-Digital Training

codingOn

OpenCV 기초

OpenCV 데이터 종류

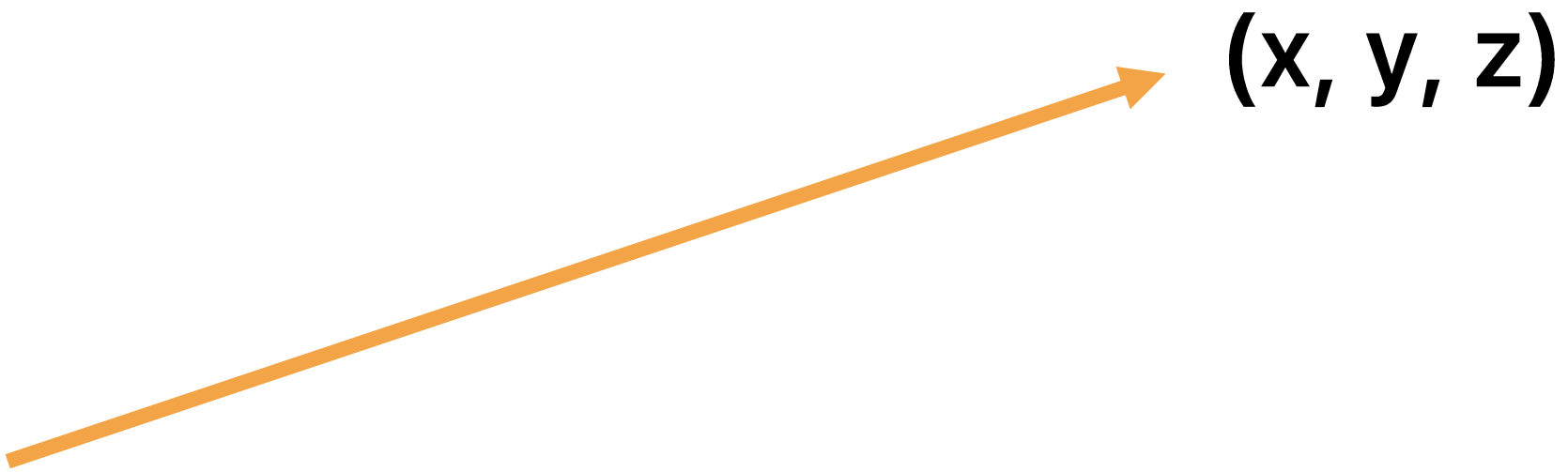
- Vector, 벡터
- Point, 포인트
- Scalar, 스칼라
- Size, 사이즈
- Range, 범위
- Rect, 직사각형
- RotatedRect, 회전 직사각형
- Mat, 2차원 배열

https://www.youtube.com/watch?v=ihNZlp7iUHE&t=64s&ab_channel=KhanAcademy

* Vector, Scalar 차이 설명

codingOn

Vector 크기와 방향을 함께 갖는 값



Vector 구조체

codingOn

OpenCV 형식	요소의 개수	데이터 타입	의미
Vec2b	2	byte	2개의 요소를 지닌 byte 벡터 구조체
Vec2w	2	ushort	2개의 요소를 지닌 ushort 벡터 구조체
Vec2s	2	short	2개의 요소를 지닌 short 벡터 구조체
Vec3i	3	int	3개의 요소를 지닌 int 벡터 구조체
Vec4f	4	float	4개의 요소를 지닌 float 벡터 구조체
Vec6d	6	double	6개의 요소를 지닌 double 벡터 구조체

- Vec<요소의 수><데이터 타입> 형태로 구성

```
Vec4d vecotr1 = new Vec4d(1.0, 2.0, 3.0, 4.0);
Vec4d vecotr2 = new Vec4d(1.0, 2.0, 3.0, 4.0);

textBox.Text = "";
textBox.Text += vecotr1.Item0.ToString() + "\r\n"; // 1
textBox.Text += vecotr1[1].ToString() + "\r\n"; // 2
textBox.Text += vecotr1.Equals(vecotr2).ToString() + "\r\n"; // True
```

Item0, Item1, ... 으로 요소 접근

codingOn

Point 크기가 없고 위치만 있는 값

- (x, y)

Point 구조체

codingOn

OpenCV 형식	요소의 개수	데이터 타입	의미
Point	2	int, double	2개의 요소를 지닌 int, double 벡터 구조체
Point2f	2	float	2개의 요소를 지닌 float 벡터 구조체
Point2d	2	double	2개의 요소를 지닌 double 벡터 구조체
Point3f	3	float	3개의 요소를 지닌 float 벡터 구조체
Point3d	3	double	4개의 요소를 지닌 double 벡터 구조체

- Point<요소의 수><데이터 타입> 형태로 구성

```
Vec3d vector = new Vec3d(1.0, 2.0, 3.0);
Point3d pt1 = new Vec3d(1.0, 2.0, 3.0); // Point에 Vector 인스턴스 생성 가능
Point3d pt2 = vector; // Point에 Vector 대입 가능

textBox.Text = pt1.ToString() + "\r\n"; // (x:1 y:2 z:3)
textBox.Text += pt2.ToString() + "\r\n"; // (x:1 y:2 z:3)
textBox.Text += pt1.X.ToString() + "\r\n"; // 1
```

X, Y, Z 로 요소 접근

Point 구조체

```
OpenCvSharp.Point pt1 = new OpenCvSharp.Point(1, 2);
OpenCvSharp.Point pt2 = new OpenCvSharp.Point(3, 4);

textBox.Text = pt1.DistanceTo(pt2).ToString() + "\r\n"; // 2
textBox.Text += pt1.DotProduct(pt2) + "\r\n"; // 7
textBox.Text += pt1.CrossProduct(pt2) + "\r\n"; // -4
textBox.Text += (pt1 + pt2).ToString() + "\r\n"; // (x:4 y:4)
textBox.Text += (pt1 - pt2).ToString() + "\r\n"; // (x:-2 y:0)
textBox.Text += (pt1 == pt2).ToString() + "\r\n"; // False
textBox.Text += (pt1 * 0.5).ToString() + "\r\n"; // (x:0 y:0)
```

- DistanceTo() : 다른 Point 까지의 거리
- DotProduct() : 두 포인트의 내적
- CrossProduct() : 두 포인트의 외적
- 비교 연산은 ==, != 만 가능

codingOn

Scalar 위치, 방향 없이 크기만 있는 값

Scalar 구조체

- 주로 칼라(RGBA)를 표현할 때 사용
- 제네릭 타입을 상속하기 때문에 벡터와 상호 캐스팅은 불가능
- 배정밀도 부동소수점(64bit)을 사용
  - float = 32bit
  - double = 64bit
- 요소의 수는 4개로 고정
  - 값을 2개만 입력할 경우 나머지 2개의 값은 0으로 채워짐

Scalar 구조체

```
Scalar s1 = new Scalar(252, 427);
Scalar s2 = Scalar.Yellow;
Scalar s3 = Scalar.All(99);
```

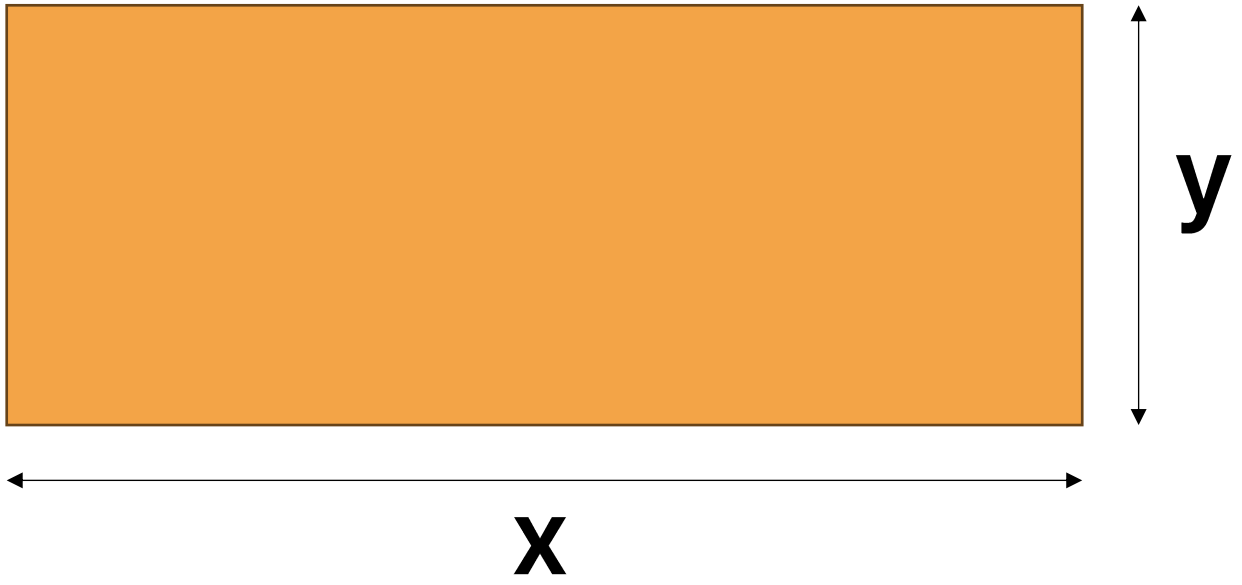
→ 각종 컬러에 대한 기본 값도 제공

```
textBox.Text = s1.ToString() + "\r\n";  // [255, 127, 0, 0]
textBox.Text += s2.ToString() + "\r\n"; // [0, 255, 255, 0]
textBox.Text += s3.ToString() + "\r\n"; // [99, 99, 99, 99]
```

연산	사용 예	반환값 예시
모든 값 할당	Scalar.All(v)	[v, v, v, v]
RGB 형식 변환	Scalar.FromRgb(r, g, b)	[b, g, r, 0]
무작위 색상	Scalar.RandomColor( )	[94, 254, 248, 0]
요서별 곱셈	s1.mul (s2)	[22320, 2673, 0, 0]
켤레	s1.Conj( )	[v, -v, v, -v]
실수 확인	s1.isReal( )	모든 값이 0일 경우 True, 아니면 False
형식 변환	s1.ToVec3b( )	벡터 구조체 Vec3b 형식으로 변환

codingOn

Size 이미지의 크기를 다룸



Size 구조체

- 이미지의 크기를 나타내는 것에 주로 사용
- 너비(Width)와 높이(Height)를 멤버로 사용
- Mat 클래스에서 Size 구조체를 메서드처럼 사용 가능

```
OpenCvSharp.Size size = new OpenCvSharp.Size(640, 480);
OpenCvSharp.Size2d size2D = new OpenCvSharp.Size2d(1280.0, 720.0); // double
OpenCvSharp.Size2f size2F = new OpenCvSharp.Size2f(320.0f, 240.0f); // float

Mat img = new Mat(size, MatType.CV_8UC3); // 8bit unsigned, 3 color

textBox.Text = $"{size.Width}, {size.Height}";                    // 640, 480
textBox.Text += img.Size();                              // (width:640 height:480)
textBox.Text += $"{img.Size().Width}, {img.Size().Height}"; // 640, 480
textBox.Text += $"{img.Width}, {img.Height}";                     // 640, 480
```

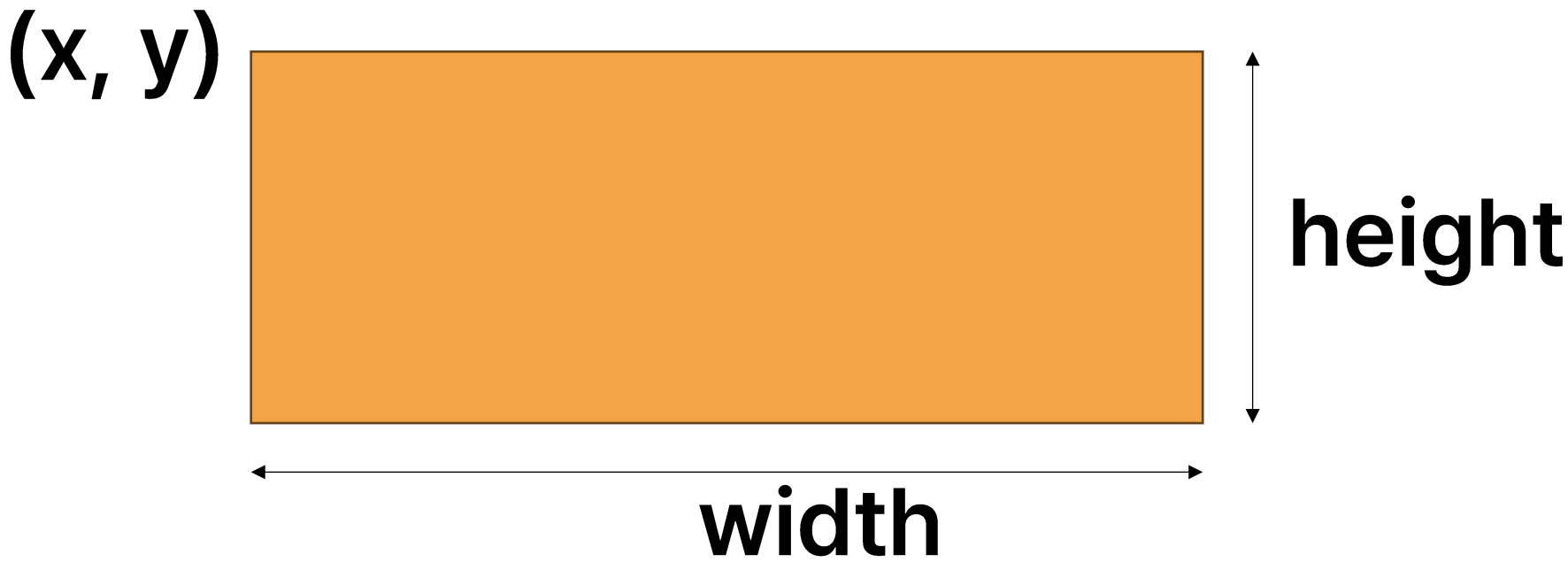
Range 구조체

- 특정 시퀀스의 범위를 지정하는데 사용
- 범위의 종료값은 포함되지 않음

```
OpenCvSharp.Range range = new OpenCvSharp.Range(0, 100);

textBox.Text = $"{range.Start}, {range.End}"; // 0, 100
```

codingOn

Rect 사격형 도형을 다룸



Rect 구조체

- 좌측 상단을 의미하는 Point 구조체를 사용
- 너비와 높이를 의미하는 Size 구조체를 사용
- 오버로드된 생성자를 통해 x, y, width, height 값으로 사각형 생성 가능

```
OpenCvSharp.Rect rect1 = new OpenCvSharp.Rect
    (new OpenCvSharp.Point(0, 0), new OpenCvSharp.Size(640, 480));

OpenCvSharp.Rect rect2 = new OpenCvSharp.Rect(10, 10, 640, 480);

textBox.Text = rect1.ToString(); // (x:0 y:0 width:640 height:480)
textBox.Text += rect2.ToString(); // (x:10 y:10 width:640 height:480)
```

Rect 구조체

codingOn

연산	사용 예	반환값 예시
멤버 접근	rect.X, rect.Y, rect.Width, rect.Height rect.Left, rect.Right, rect.Top, rect.Bottom	int
	rect.TopLeft, rect.BottomRight	Point
좌측 상단 지점	rect.Location	Point
특정 위치가 직사각형 내부에 있는지 확인	rect.Contains(Point)	Boolean
두 직사각형의 영역 합집합	rect1.Union(rect2)	Rect
두 직사각형의 영역 교집합	rect1.Intersect(rect2)	Rect
직사각형 팽창	rect.Inflate(Size)	rect의 값을 직접 변형

Rect 구조체

- 직사각형 관련 연산

연산	예제
직사각형을 Point 만큼 이동	Rect = Rect + Point 또는 Rect += Point Rect = Rect - Point 또는 Rect -= Point
직사각형 Size 만큼 확대	Rect = Rect + Size Rect += Size
직사각형 Size 만큼 축소	Rect = Rect - Size Rect -= Size
두 직사각형 영역의 합집합	Rect = Rect1 | Rect 2 또는 Rect1 |= Rect2
두 직사각형의 영역 교집합	Rect = Rect1 & Rect2 또는 Rect1 &= Rect2
두 직사각형의 같음 비교	Boolean = Rect1 == Rect2
두 직사각형의 다름 비교	Boolean = Rect1 != Rect2

RotatedRect 구조체

- 직사각형에 중심점, 크기, 각도가 추가됨
- 중심점을 기준으로 크기를 지정하고 회전
- float 형태의 데이터를 사용

연산	사용 예	반환값 예시
멤버에 접근	rotatedRect.Center rotatedRect.Size rotatedRect.Angle	Point2f Size2f float
회전된 직사각형을 포함하는 직사각형	rotatedRect.BoundingRect( )	Rect
회전된 직사각형의 4개의 코너	rotatedRect.Points( )	Point2f[ ]

RotatedRect 구조체

codingOn

연산	사용 예	반환값 예시
멤버에 접근	rotatedRect.Center rotatedRect.Size rotatedRect.Angle	Point2f Size2f float
회전된 직사각형을 포함하는 직사각형	rotatedRect.BoundingRect( )	Rect
회전된 직사각형의 4개의 코너	rotatedRect.Points( )	Point2f[ ]

```
RotatedRect rotatedRect = new RotatedRect
    (new Point2f(100f, 100f), new Size2f(100f, 100f), 45f);

textBox.Text = rotatedRect.BoundingRect().ToString() + "\r\n"; // (x:29 y:29 width:143 height:143)
textBox.Text += rotatedRect.Points().Length.ToString() + "\r\n"; // 4
textBox.Text += rotatedRect.Points()[0].ToString() + "\r\n"; // (x:29.289322 y:100.00001)
```

codingOn

Mat 2차원 행렬

23	14	76	34	21	86
12	8	32	12	43	30
…					

row

col

Mat 데이터 클래스

- OpenCV에서 **가장 많이 사용**하는 클래스
- 주로 **이미지를 저장**하는데 사용
- **레스터 주사(Raster Scan) 방식**으로 데이터를 저장하는 **2차원 배열**

0	1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24	25
26	27	28	29	30	...							

Mat 데이터 클래스

- 생성자를 통해 Mat(행렬) 생성이 가능
- Create( )를 사용한 동적 생성도 가능

```
// 640 x 480 크기의 3ch(채널) 행렬 만들기
Mat m1 = new Mat(480, 640, MatType.CV_8UC3);
Mat m2 = new Mat(new OpenCvSharp.Size( 640, 480), MatType.CV_8UC3);

Mat M = new Mat();

// 세 개 모두 같은 의미
M.Create(MatType.CV_8UC3, new int[] { 640, 480 });
// M.Create(new OpenCvSharp.Size(640, 480), MatType.CV_8UC3);
// M.Create(480, 640, MatType.CV_8UC3);

M.SetTo(new Scalar(255, 0, 0)); // (255, 0, 0)으로 행렬의 모든 요소를 초기화
```

480, 640

Mat 데이터 클래스

- Mat 요소에 직접 접근하여 값을 가져오려면 **At( ), Get( )** 매서드를 사용

```
// Mat.Eye( ) 단위 행렬 생성, 단위 행렬: 대각선의 요소만 1이고 나머지는 0인 행렬
// 3 x 3, 64bit double 3ch 행렬 생성
Mat m = Mat.Eye(new OpenCvSharp.Size(3, 3), MatType.CV_64FC3);

textBox.Text = m.At<double>(0, 0).ToString() + "\r\n";      // 1

// Vector로 받아올 경우 첫 번째 요소에만 값이 들어감
textBox.Text += m.At<Vec3d>(0, 0).Item0.ToString() + "\r\n";// 1
textBox.Text += m.At<Vec3d>(1, 1).Item1.ToString() + "\r\n";// 0
textBox.Text += m.At<Vec3d>(2, 2).Item2.ToString() + "\r\n";// 0

// Point도 Vector 처럼 첫 번째 요소에만 값이 들어감 (x:1, y:0, z:0)
textBox.Text += m.At<Point3d>(2, 2).ToString() + "\r\n";

// 형식이 맞지 않는 경우 이상값이 들어감 4607182418800017408
textBox.Text += m.At<long>(2, 2).ToString() + "\r\n";
```

Mat 데이터 클래스

- 값을 덮어 쓰려면 **Set( )** 매서드를 사용
  - Set( )은 type과 value의 형식을 일치시켜야 함

메서드	설명
m.At<type>(i)	type 형식 행렬 m의 i 요소 반환
m.At<type>(i, j)	type 형식 행렬 m의 i, j 요소 반환
m.At<type>(i, j, k)	type 형식 행렬 m의 i, j, k 요소 반환
m.At<type>(idx)	type 형식 행렬 m의 int [ ] 가 가리키는 N차원 요소 반환
m.Set<type>(i, value)	type 형식 행렬 m의 i 요소를 value로 설정
m.Set<type>(i, j, value)	type 형식 행렬 m의 i, j 요소를 value로 설정
m.Set<type>(i, j, k, value)	type 형식 행렬 m의 i, j, k 요소를 value로 설정
m.Set<type>(idx, value)	type 형식 행렬 m의 int[ ]가 가리키는 N차원 요소를 value로 설정

Mat 데이터 클래스 - 요소 접근

codingOn

* m이 Mat 클래스의 인스턴스 일 때,
** 모든 Row는 Col로 바꾸어 행이 아닌 열을 기준으로 데이터 접근 가능

메서드	설명
m.Row.Get(i)	행렬 m의 행 i에 해당하는 배열 반환
m.Row.Get(i0, i1) m.RowRange(i0, i1)	행렬 m의 행 i0 ~ (i1 - 1)에 해당하는 배열 반환
m.Row.Get(Range) m.RowRange(Range)	행렬 m의 행 범위 구조체에 해당하는 배열 반환
m.Row.Set(i, Mat)	행렬 m의 행 i에 해당하는 배열을 Mat 배열로 변경
m.Row.Set(i0, i1, Mat)	행렬 m의 행 i0 ~ (i1 - 1)에 해당하는 배열을 Mat 배열로 변경
m.Row.Set(Range, Mat)	행렬 m의 범위 구조체에 해당하는 배열을 Mat 배열로 변경

Mat 데이터 클래스 - 요소 접근

codingOn

* m이 Mat 클래스의 인스턴스 일 때,

메서드	설명
m[Rect]	행렬 m의 직사각형 구조체에 해당하는 배열 반환
m[Range[ ]]	행렬 m의 범위 구조체에 해당하는 배열 반환
m[Range, Range]	행렬 m의 범위 구조체에 해당하는 배열 반환
m[i0, i1, j0, j1]	행렬 m의 행 i0~(i1 - 1), j0 ~ (j1 - 1)에 해당하는 배열 반환

이미지 불러오기

- Cv2.ImRead("이미지 파일 경로 및 이름", 플래그 변수)

```
// ImreadModes.ReducedColor2 : 이미지 크기를 1/2로 줄이고 컬러로 가져오기
Mat src = Cv2.ImRead("elden_ring.jpg", ImreadModes.ReducedColor2);

textBox.Text = src.ToString();
// Mat [ 360*640*CV_8UC3, IsContinuous=True, IsSubmatrix=False,
//       Ptr=0x2d10b7e3a90, Data=0x2d102c1be40 ]
```

이미지 불러오기

- 대표적인 플래그 변수 종류
  - ImreadModes.Unchanged : 변화 없음
  - ImreadModes.Grayscale : 1채널 그레이 스케일
  - ImreadModes.Color : 다중 채널 컬러
  - ImreadModes.ReducedColor2 : 크기를 1/2로 줄인 컬러
  - ImreadModes.ReducedGrayscale4 : 크기를 1/4로 줄인 그레이 스케일

이미지 출력하기

- Cv2.ImShow("창 이름", Mat 인스턴스)
  - 이미지를 표시하는 새 창을 띄움

```
// ImreadModes.ReducedColor2 : 이미지 크기를 1/2로 줄이고 컬러로 가져오기
Mat src = Cv2.ImRead("elden_ring.jpg", ImreadModes.ReducedColor2);

Cv2.ImShow("image window", src);

textBox.Text = src.ToString();
// Mat [ 360*640*CV_8UC3, IsContinuous=True, IsSubmatrix=False,
//       Ptr=0x2d10b7e3a90, Data=0x2d102c1be40 ]
```

이미지 출력하기

- OpenCvSharp.WpfExtensions.
  BitmapSourceConverter.ToBitmapSource(Mat)
  - Mat의 이미지 데이터를 WPF의 Image 컨트롤에 표시하기 위해 Bitmap으로 변환

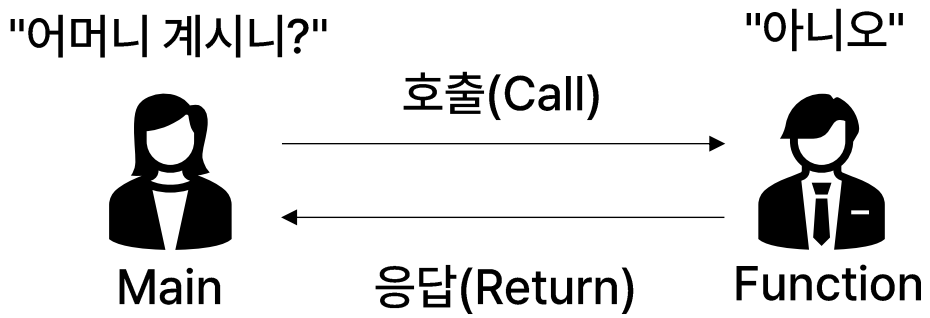
```
// ImreadModes.ReducedColor2 : 이미지 크기를 1/2로 줄이고 컬러로 가져오기
Mat src = Cv2.ImRead("elden_ring.jpg", ImreadModes.ReducedColor2);

Cv2.ImShow("image window", src);

imageBox.Source = OpenCvSharp.WpfExtensions.BitmapSourceConverter.ToBitmapSource(src);
```

마우스 콜백

- 마우스 이동, 버튼 입력에 대한 정보를 받아오는 용도
- MouseCallback 인스턴스를 만들고 이벤트 메소드 등록
- Cv2.SetMouseCallback으로 이벤트를 콜백으로

참고~ 콜백이란?

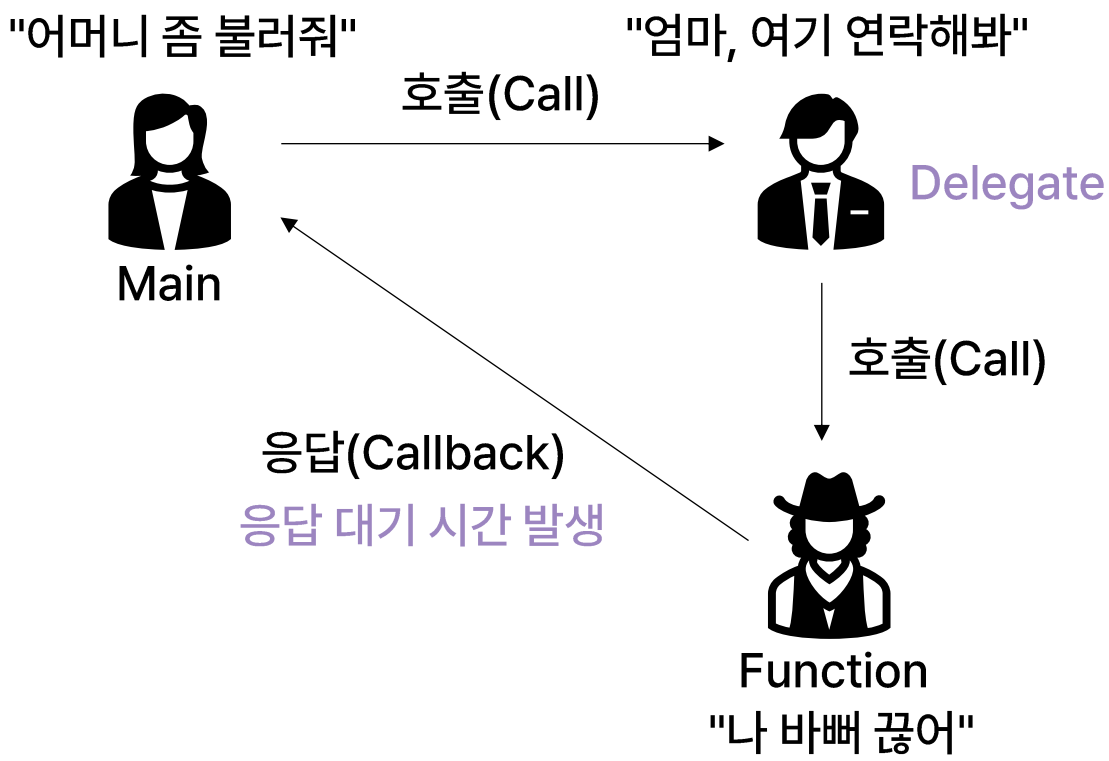
- 일반적인 Call



* Delegate는 언제 사용?

다른 인스턴스에 변수, 배열 같은 값이 아니라
코드 자체를 전달하고 싶을 때 사용

- Callback (with Delegate)



참고~ Delegate와 Event의 차이점

codingOn

1. 이벤트는 interface 내부에서 선언할 수 있지만, delegate는 선언할 수 없음
2. 이벤트는 public으로 선언되어 있어도 자신이 선언되어 있는 클래스 외부에서 호출 할 수 없음
   - Invoke를 통한 호출만 가능

- delegate : Callback 용도
- event : 객체의 상태 변화, 사건의 발생을 알리는 용도, Callback으로 사용하기도 함

참고~ Event 사용 예시

- https://www.hind.pe.kr/1137

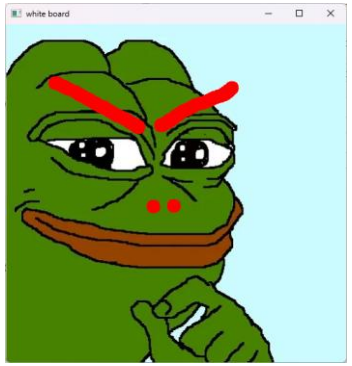
마우스 콜백 (이벤트)

```
Mat src = new Mat(new OpenCvSharp.Size(500, 500), MatType.CV_8UC3, new Scalar(255, 255, 255));

Cv2.ImShow("white board", src);

MouseCallback cvMouseCallback = new MouseCallback(MyMouseEvent);
Cv2.SetMouseCallback("white board", cvMouseCallback, src.CvPtr);
Cv2.WaitKey();
Cv2.DestroyAllWindows();
```

마우스 작동에 대한 메소드를 이벤트로 등록

```
1 reference
static void MyMouseEvent(MouseEventTypes @event, int x, int y, MouseEventFlags flags, IntPtr userdate)
{
    if(flags == MouseEventFlags.LButton)
    {
        MessageBox.Show($"마우스 좌표 {x}, {y}");
    }
}
```

실습1. 이미지 위에 마우스로 그리기

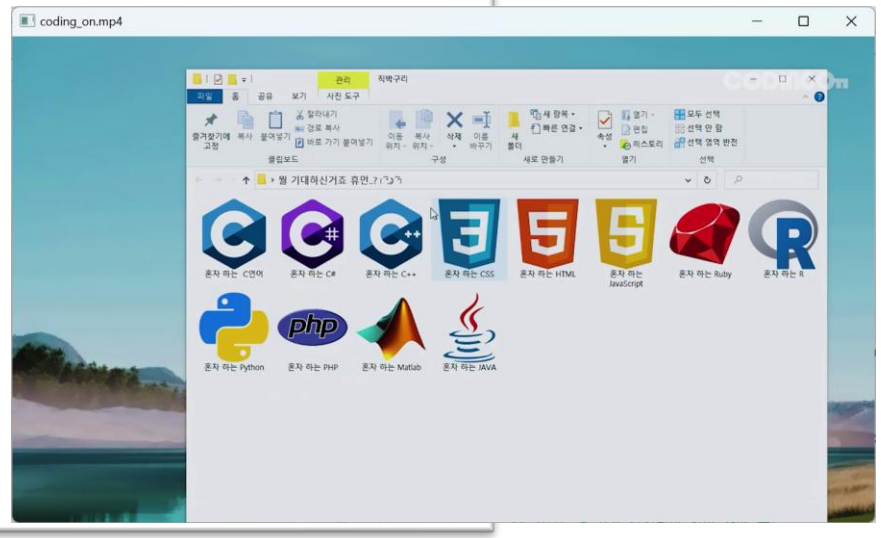


- MouseEventCallback 사용
- Event 메소드의 입력값인 마우스 좌표 활용
- 이미지 위에 마우스 왼쪽 클릭시 그림 그릴 수 있도록
- Cv2.Circle( ) 활용
  - https://shimat.github.io/opencvsharp_docs/html/a8e55867-6258-78ac-20d4-d25d8b973391.htm
  - https://shalchoong.tistory.com/26
- 힌트, 작은 Circle 사용

동영상 출력하기

- VideoCapture 클래스

```
VideoCapture capture = new VideoCapture("coding_on.mp4");
Mat frame = new Mat();

while(true)
{
    if (capture.PosFrames == capture.FrameCount) capture.Open("coding_on.mp4");

    capture.Read(frame);
    Cv2.ImShow("coding_on.mp4", frame);

    // 키보드 q 키가 감지되면 영상 재생 종료
    if (Cv2.WaitKey(33) == 'q') break;
}

// 메모리를 매우 많이 차지하기 때문에 꼭 릴리즈 해주기
capture.Release();
Cv2.DestroyAllWindows(); // 모든 OpenCV 창 닫기
```



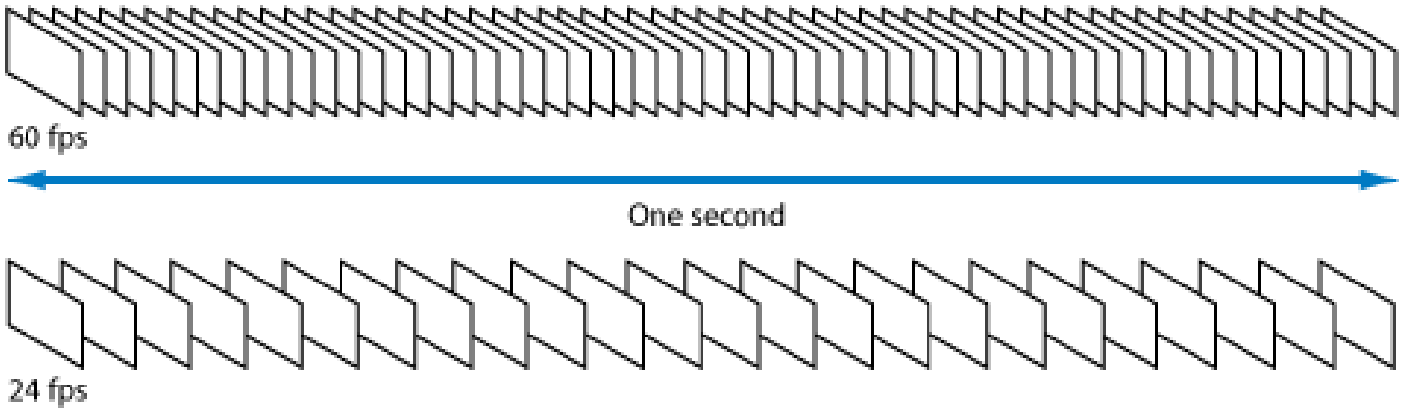
동영상 출력하기

- 동영상 재생 구조

```
// capture.PosFrames : 현재 프레임 수
// capture.FrameCount : 총 프레임 수
// 두 개가 같음 = 영상이 끝남 = while 문이 멈추지 않으면 Error 발생
// 본 코드에서는 Break 대신 영상 파일을 다시 Open하여 처음부터 다시 재생
if (capture.PosFrames == capture.FrameCount) capture.Open("coding_on.mp4");

// 동영상 파일에서 프레임 하나를 가져와 Mat에 할당
capture.Read(frame);
Cv2.ImShow("coding_on.mp4", frame);

// 키보드 q 키가 감지되면 영상 재생 종료
// 더불어 33ms를 대기 = 영상이 30fps이기 때문에 약 33ms 간격으로
// 다음 프레임을 재생 시 정상적인 속도로 재생됨
if (Cv2.WaitKey(33) == 'q') break;
```

FPS(Frame Per Second)

- 초당 프레임 수
  - 1초에 30 프레임으로 구성된 동영상 = 30 FPS
  - **FPS = 1000(ms) / 프레임 간 시간 간격(ms)**
    - 1000(ms) / 33(ms) = 약 30
    - 프레임을 33ms 마다 변경하면 30 FPS 영상이 됨



FPS(Frame Per Second)

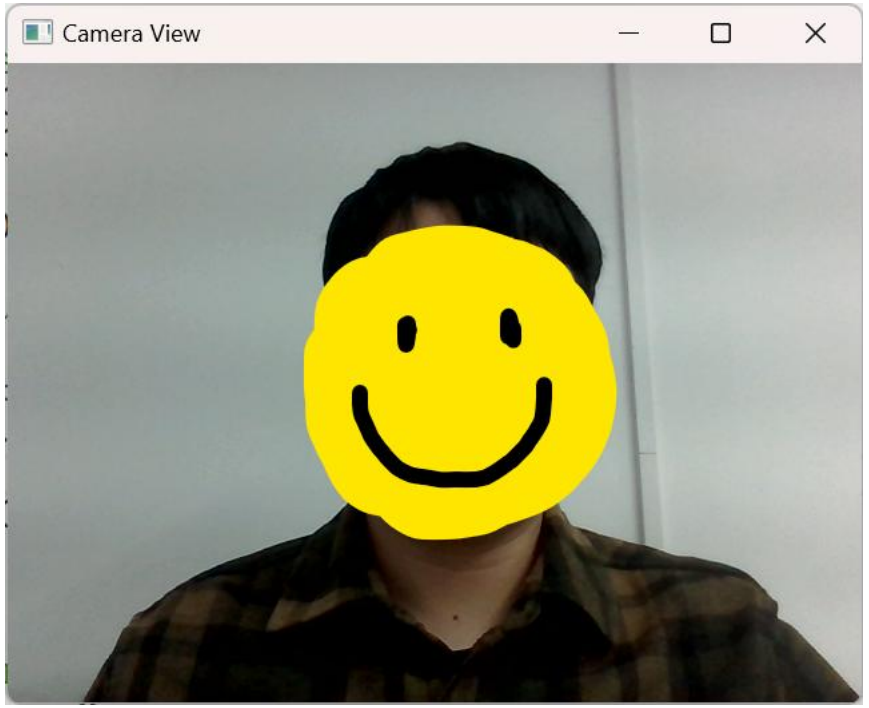
codingOn

- VideoCapture 클래스의 Get( ) 메소드의 매개변수로 **VideoCaptureProperties** 열거형을 전달하여 FPS를 포함한 동영상의 각종 정보를 받아옴

```
VideoCapture capture = new VideoCapture("coding_on.mp4");

double time_stemp = capture.Get(VideoCaptureProperties.PosMsec);
double frame_now = capture.Get(VideoCaptureProperties.PosFrames);
double frame_width = capture.Get(VideoCaptureProperties.FrameWidth);
double frame_height = capture.Get(VideoCaptureProperties.FrameHeight);
double frame_count = capture.Get(VideoCaptureProperties.FrameCount);
double fps = capture.Get(VideoCaptureProperties.Fps);
```

카메라 출력하기

```
// 0: 연결된 디바이스 id, 윈도우에서 지정함, 0, 1, 2, 3 ...
VideoCapture capture = new VideoCapture(0);
Mat frame = new Mat();

// 카메라 영상 해상도 설정
capture.Set(VideoCaptureProperties.FrameWidth, 640);
capture.Set(VideoCaptureProperties.FrameWidth, 480);

while (true)
{
    // 카메라 장치와 연결이 유지되고 있다면
    if(capture.IsOpened() == true)
    {
        capture.Read(frame);
        Cv2.ImShow("Camera View", frame);

        if (Cv2.WaitKey(33) == 'q') break;
    }
}

// 카메라 장치와 연결 해제
capture.Release();
Cv2.DestroyAllWindows(); // 모든 OpenCV 창 닫기
```



실습2. 마우스 포인터 위치 표시

- 카메라 영상 위에 마우스 포인터를 올리고 왼쪽 클릭을 누르고 있으면, 커서 위치가 빨간 원으로 표시되도록 만들기
- 실습1번 문제 응용