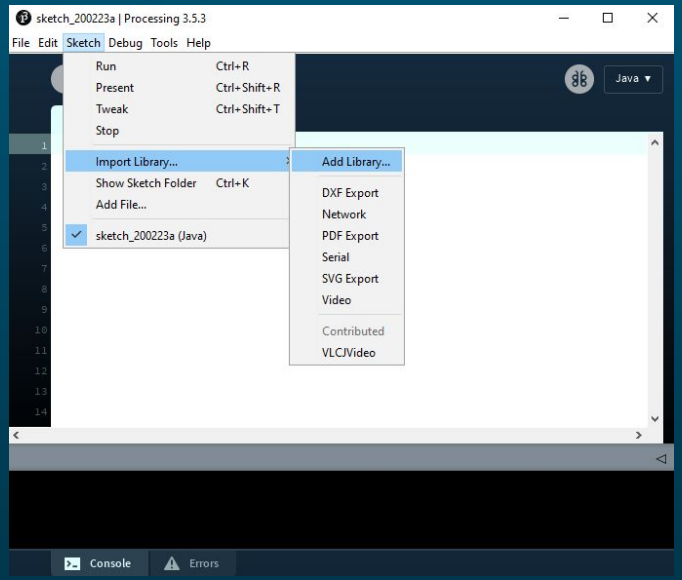
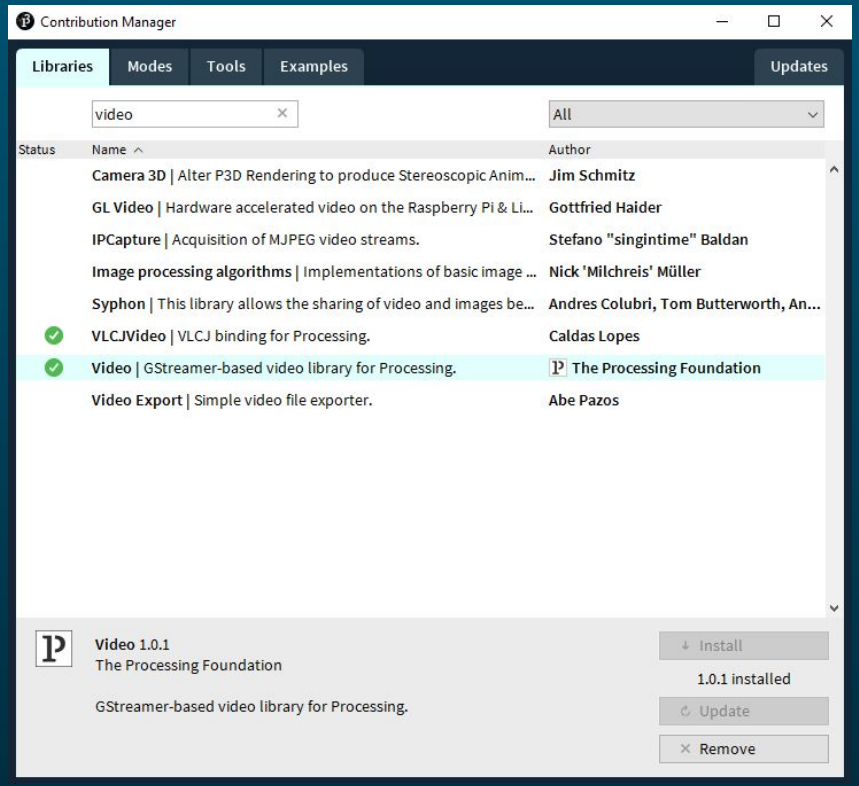
Week 05
Video Processing

by Mike Wong. School of Creative Media

Install Processing Video Library





Movie class of Video Library

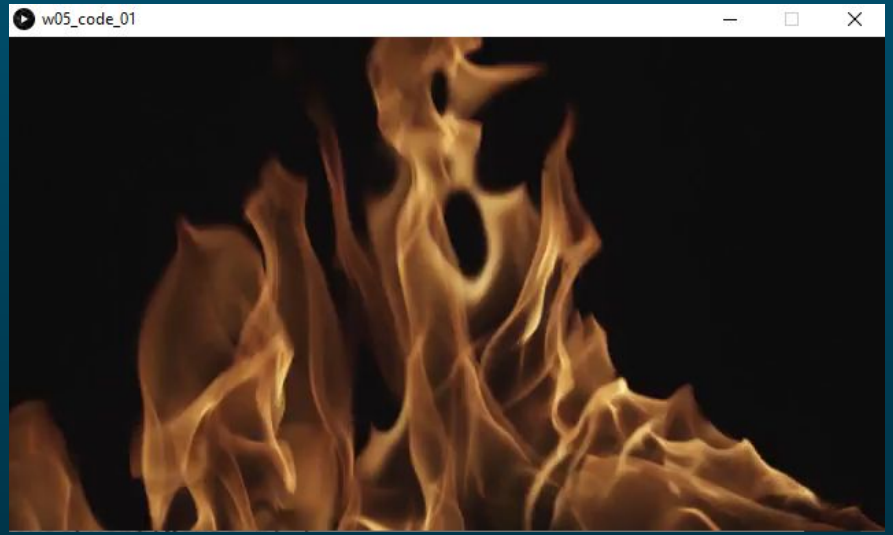
Methods	Description
`.read()`	Reads the current frame
`.play()`	Plays the movie
`.pause()`	Pauses the movie
`.stop()`	Stops the movie
`.loop()`	Plays the movie looped
`.noLoop()`	Cancels looping
`.available()`	Returns TRUE if a new movie frame is ready.

Methods	Description
`.jump()`	Jumps to the specified time
`.duration()`	Returns the movie duration
`.time()`	Returns time of current frame
`.speed()`	Sets the movie speed (default= 1.0)
`.frameRate()`	Sets the playback fps rate

Example 01 - Simple playback

w05_code_01

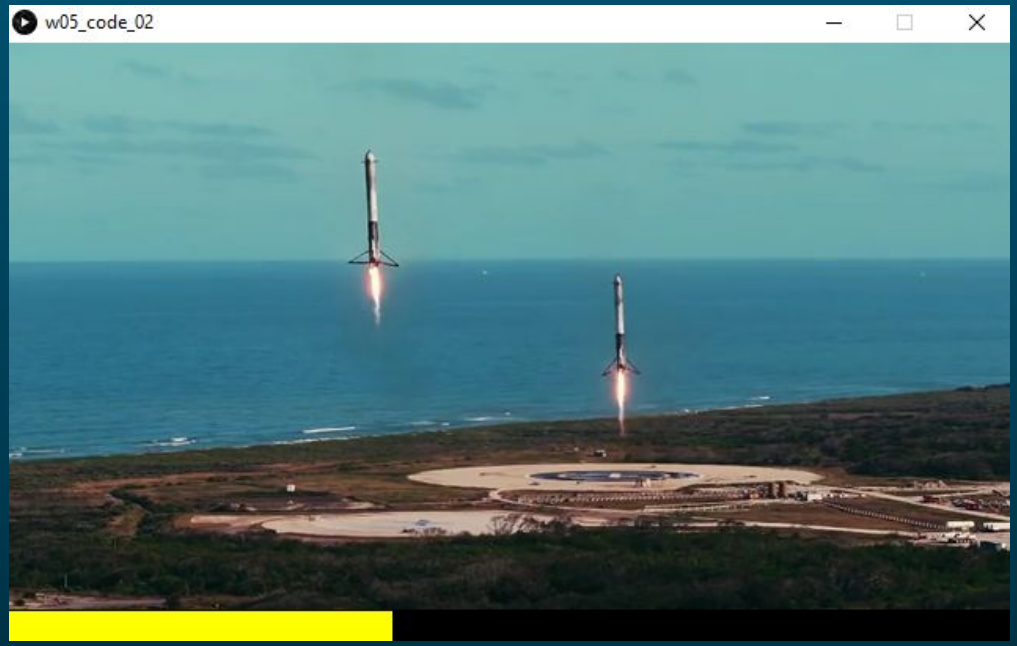
```
import processing.video.*;
Movie  mov;

void setup() {

  size(640, 360);
  mov = new Movie(this, "fire_360.mp4");
  //mov.play();  //  Play the movie once
  mov.loop();  //  Loop the video

}

void draw() {
  image(mov,0,0);
}

//  The following function is called whenever a new frame
//  from the movie becomes available
void movieEvent(Movie m) {
  m.read();
}
```



Example 02 - interaction via `.jump()`

w05_code_02

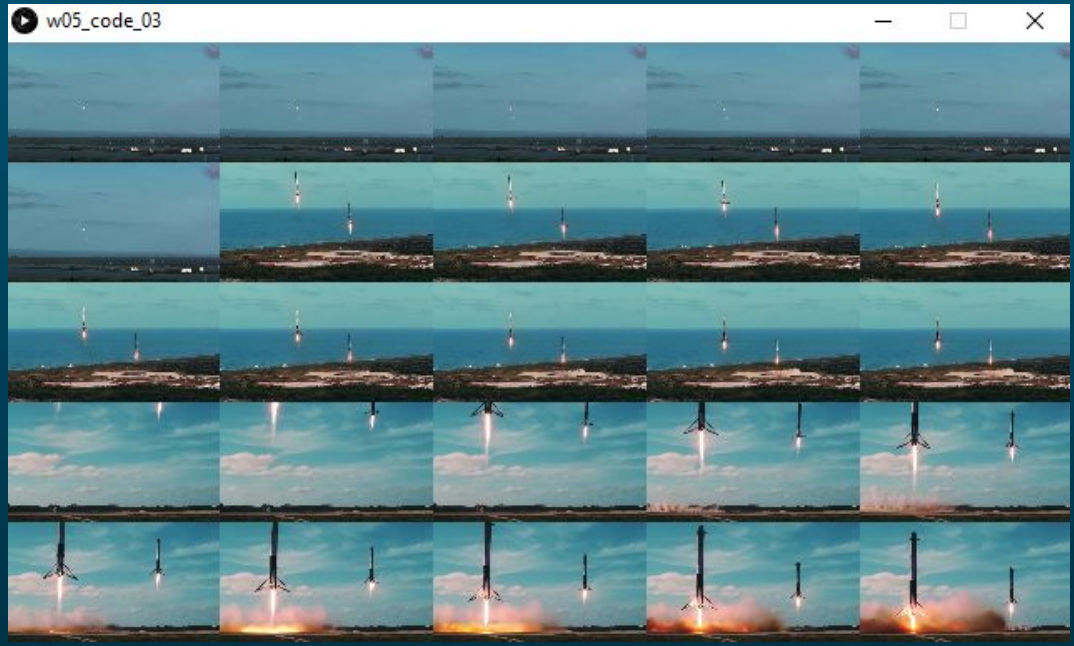
```
import processing.video.*;
Movie  mov;
float movLength;  // Length of movie in seconds.

void setup() {
  size(640, 380);
  background(0);
  println("Loading movie");
  mov = new Movie(this, "falcon.mp4");
  println("Movie loaded");
  mov.loop();  //  MUST PLAY it !
  movLength = mov.duration();
  fill(255,255,0);  //  timeline color
}

void draw() {
  background(0);
  if (mousePressed) {  //  Timeline Drag
    float now = map(mouseX, 0,width, 0,movLength);
    rect(0,360,mouseX,20);
    mov.jump(now);
  }
  else {
    float now = map(mov.time(), 0,movLength, 0, width);
    rect(0,360, now,20);
  }
  image(mov, 0,0);  //  Display Movie frame
}
```



Example 03 - Contact Sheet

w05_code_03

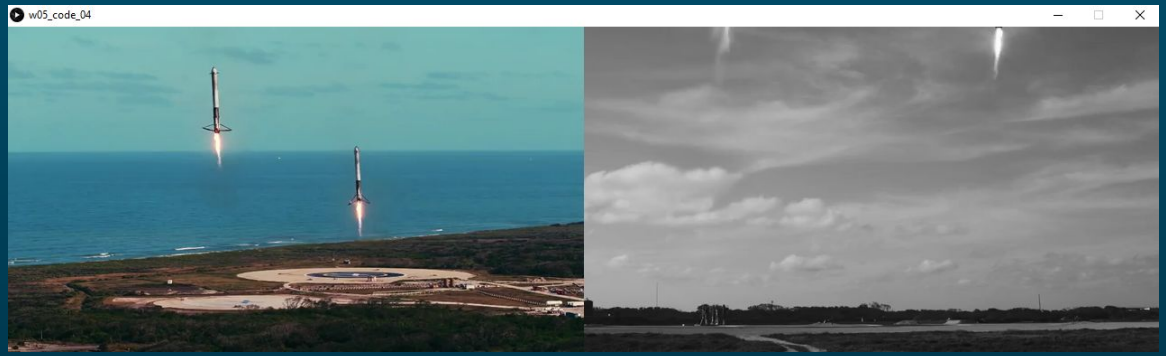
```
import processing.video.*;
Movie  mov;

//  Grid-related
int numDiv = 5;
int numBlocks = numDiv * numDiv;
int bw, bh;
int counter = 0;

//  Create uniform timemarks
float[] timeMark = new float[numBlocks];
float movLength;

void setup() {
  size(640, 360);
  background(0);
  println("Loading movie");
  mov = new Movie(this, "falcon.mp4");
  println("Movie loaded");
  mov.play();  //  MUST PLAY it !

  bw = width/numDiv;
  bh = height/numDiv;
  numBlocks = numDiv * numDiv;

  movLength = mov.duration();
  for (int i = 0; i < numBlocks; i++) {
    timeMark[i] = map(i, 0,numBlocks, 0,movLength);
  }
}

void draw() {
  mov.jump(timeMark[counter]);
  int x = counter % numDiv;
  int y = counter / numDiv;
  int px = int(map(x, 0, numDiv, 0, width));
  int py = int(map(y, 0, numDiv, 0, height));
  image(mov, px,py, bw,bh);
  counter++;
  if (counter >= numBlocks) {
    counter = 0;
  }
}
```



Example 04 - Multi-movies

w05_code_04

```
1  import processing.video.*;
2
3  Movie  movL, movR;
4
5  void setup() {
6
7    size(1280, 360);
8    background(0);
9    println("Loading movie");
10   movL = new Movie(this, "falcon.mp4");
11   movR = new Movie(this, "falcon.mp4");
12   println("Movie loaded");
13
14   movL.loop();
15   movR.loop();
16   movR.speed(3.0);
17
18 }
19
20 void draw() {
21   image(movL,   0,0);
22   PImage bw = movR.copy();
23   bw.filter(GRAY);
24   image(bw, 640,0);
25 }
```



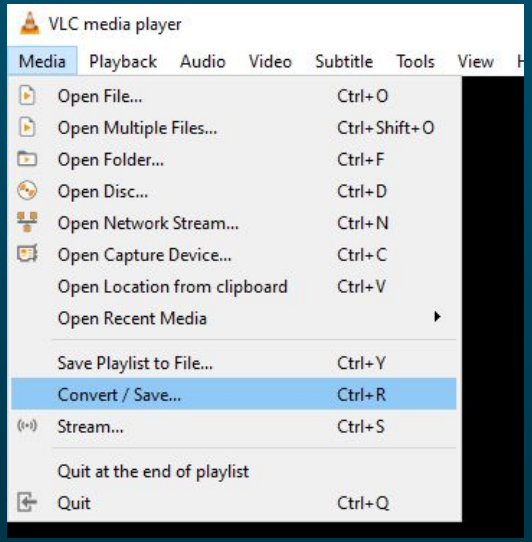
About `.speed()` of `Movie` class

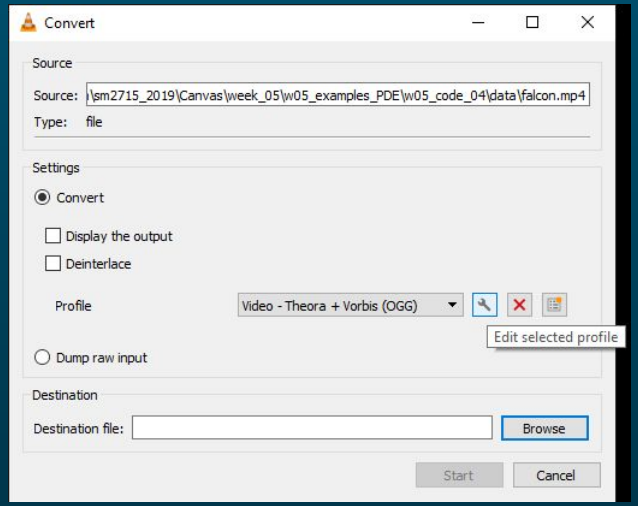
```
.speed(<param>);
```

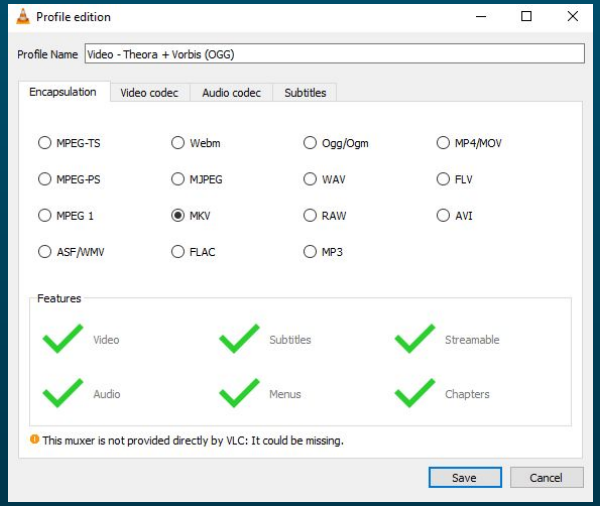
accepts a floating point number as parameter, and the default speed is `1.0`. It also accepts a **negative number** for realising **reversed playback** but **its current implementation seems to be incomplete**. Not all movies support reversed playback, the ones encoded in Theora codec will work.

```
// Reverse the playback
mov.speed(-1.0);
```

VLC for encoding Theora encoded .mkv







Example 05 - Reversed Playback*

w05_code_05

```
import processing.video.*;
Movie  mov;

void setup() {

  size(640, 360);
  background(0);
  println("Loading movie");
  mov = new Movie(this, "falcon_theora.mkv");
  println("Movie loaded");

  mov.play();
  revPlay(mov);
}

void draw() {
  set(0,0, mov);
}

void revPlay(Movie m) {
  m.speed(1.0);
  m.jump(m.duration());
  m.speed(-1);
}
```

Capture class of Video Library

Methods	Description
`.available()`	Returns TRUE if a new frame is available
`.start()`	Starts capturing
`.stop()`	Stops capturing
`.read()`	Reads the current frame
`.list()`	Gets a list of available Image Capture devices.

Example 06 - Simple WebCam

w05_code_06

```
import processing.video.*;
Capture cam;

void setup() {
  size(640, 480);
  cam = new Capture(this, width, height);
  cam.start();
}

void draw() {
  set(0,0, cam);
}

//  The following function is called whenever a new frame
//  from the camera becomes available
void captureEvent(Capture c) {
  c.read();
}
```

Example 07 - Pixelated WebCam

w05_code_07

```
import processing.video.*;
Capture cam;

void setup() {
  size(640, 480);
  cam = new Capture(this, width, height);
  cam.start();
  noStroke();
}

void draw() {
  PImage f = cam.copy();
  for (int y = 0; y < 480; y+=10) {
    for (int x = 0; x < 640; x+=10) {
      color c = f.get(x,y);
      fill(c);
      rect(x,y,10,10);
    }
  }
}

//  The following function is called whenever a new frame
//  from the camera becomes available
void captureEvent(Capture c) {
  c.read();
}
```


by Mike Wong. School of Creative Media

In-class exercise

Your sketch should load and loop a movie. When the user clicks on the display window, the current displayed content should then be used as a grayscale background image. The user may further click, and additional grayscale images may then be blended on top to the background.