3. Bluetooth og Processing

Hva er Processing?

Processing er et programmeringsspråk basert på Java. Det gjør det lett å utvikle grafiske programmer raskt.

For å vite mer, lære mer, se på eksempler av bruk og mer, besøk processing.org.

Processing i app, Processing.js

For å få til kommunikasjon over bluetooth på mobile enheter på en enkel måte, har vi laget en app. Denne appen tar inn Processing råkode, oversetter den til Javascript ved hjelp av Processing.js, og injiserer et objekt i koden dere har skrevet som gjør det mulig for dere å enkelt kommunisere med bluetooth enheten. Koden fører teknisk sett til tegning på et html canvas, og dere kan få tilgang til størrelsen på dette under injiseringa.

Sikkerhetshull

Appen oversetter til Javascript, noe som gjør at den som har skrevet koden dere henter har mulighet til mye rart. Vi har prøvd å forhindre at det er tilgjengelig alt for mange detaljer om telefonen i appen, men det er mulig å hente ut info som ikke er tilgjengelig via bare det injiserte objektet.

Vi råder derfor alle til å KUN KJØRE KODE SOM DERE STOLER PÅ! Det vil si kode som ligger hos en bruker dere har kontroll på. Som standard ser appen etter filer på hjemmeområdet ditt på NTNU. Plasseringen er folk.ntnu.no/[BRUKERNAVN]/plab/plab.pde, men dette kan endres i innstillinger i appen (trykk på det lille tannhjulet i høyre hjørnet). Hvis du ikke kjenner koden fra før, anbefaler vi at du sjekker på nett hva den faktisk er før du kjører den.

Det injiserte objektet

Javascript objektet som injiseres har tydelig definerte interfaces, som dokumenteres i InterfacesInc eksempelet på GitHub kontoen vår. I denne koden ligger også to event klasser og lytte interfacer. Denne koden er med i appen, så du trenger ikke å laste opp denne filen når du laster opp kode til appen.

InterfacesInc

```
/**
 * PLabOrientationEvent is the event object received when phone changes orientation
 */
class PLabOrientationEvent {
  /**
   * alpha - magnetic direction (in degrees)
   */
  float alpha;
  /**
   * beta - tilt front-to-back, front is positive (in degrees)
   */
  float beta;
  /**
   * gamma - tilt left-to-right, right is positive(in degrees)
   */
  float gamma;
}

/**
 * PLabAccelerationEvent is the event object received at regular intervals, telling what acceleration is
 */
class PLabAccelerationEvent {
  /**
   * x - acceleration in x direction
   */
  float x;
  /**
   * y - acceleration in y direction
   */
  float y;
  /**
   * z - acceleration in z direction
   */
  float z;
  /**
   * timestamp - when the acceleration was meassured
   */
  float timestamp;
}
```

```
/**
 * PLabBridge is the interface we have for communicating with the plab app.
 */
interface PLabBridge {
  /**
   * gets the width we have available to draw on
   */
  public int getWidth ();
  /**
   * gets the height we have available to draw on
   */
  public int getHeight ();
  /**
   * sends a string to a connected bluetooth device. Can not send more than 20 characters.
   */
  public void send (String string);
  /**
   * register callback function that will get the data sent from the connected device
   */
  public void subscribeMessages (PLabRecv sub);
  /**
   * Disconnects the device and returns to main menu
   */
  public void disconnect();
  /**
   * hides the back button
   */
  public void hideBackButton();
  /**
   * display the back button
   */
  public void showBackButton();

  /**
   * make the device vibrate for given amount of time. On iOS time will be ignored.
   */
  public void vibrate(int milliseconds);
  /**
   * Listen for device orientation changes
   */
  public void addDeviceOrientationListener(PLabOrientationListener listener);
  /**
   * remove device orientation changes listener
   */
  public void removeDeviceOrientationListener(PLabOrientationListener listener);
  /**
   * Listen for device acceleration changes
   */
  public void addDeviceAccelerationListener(PLabAccelerationListener listener);
  /**
   * remove device acceleration changes listener
   */
  public void removeDeviceAccelerationListener(PLabAccelerationListener listener);
  /**
   * Set milliseconds between each acceleration update
   */
  public void setDeviceAccelerationUpdateInterval(int milliseconds);
}

/**
 * A simple interface that defines the callback read function
 */
interface PLabRecv {
  /**
   * The callback function. Will be called when the connected device sends something to this program.
   */
  public void receive(String message);
}

/**
 * PLabOrientationListener is the interface describing an orientation change listener
 */
interface PLabOrientationListener {
  public void deviceOrientation(PLabOrientationEvent event);
}
```

```
/**
 * PLabAccelerationListener is the interface describing an acceleration event listener
 */
interface PLabAccelerationListener {
  public void deviceAcceleration(PLabAccelerationEvent event);
}
```

Oversikt over de ulike klassene og interfacene:

type	navn	beskrivelse
`class`	`PLabOrientationEvent`	Objekt som inneholder informasjon om orientering. Mottas av lytter til orienteringsendringer.
`class`	`PLabAccelerationEvent`	Objekt som inneholder informasjon om enhetens akselerasjon. Mottas av lytter til akselerasjonsendringer.
`interface`	`PLabBridge`	Det injiserte objektet. Brukes til å kommunisere med telefonen. Kopi av dette mottas en gang i metoden bindPLabBridge().
`interface`	`PLabRecv`	Inneholder en callback funksjon som mottar tekst. Implementer denne for å lytte etter meldinger som kommer fra Bluetooth enhet.
`interface`	`PLabOrientationListener`	Lytter etter endringer i orientering. Bruk denne for å følge med på hvilken vei telefonen peker.
`interface`	`PLabAccelerationListener`	Lytter etter akselerasjon. Bruk denne for å følge med på bevegelser av telefonen.

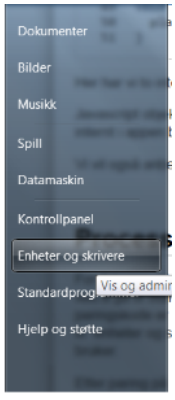
Javascript objektet injiseres ved at bindPLabBridge kalles. Her kan du registrere lyttere ved hjelp av callback funksjoner. Den som er relevante er subscribeMessages(), addDeviceOrientationListener() og addDeviceAccelerationListener().

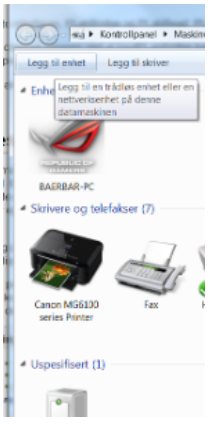
Se "Processing, Arduino, Bluetooth" for en oversikt over eksempler i bruk.

Vi vil også anbefale dere å kalle size(bridge.getWidth (), bridge.getHeight ()); i løpet av bindPLabBridge(). Da får dere muligheten til å tegne på hele skjermen til den mobile enheten.

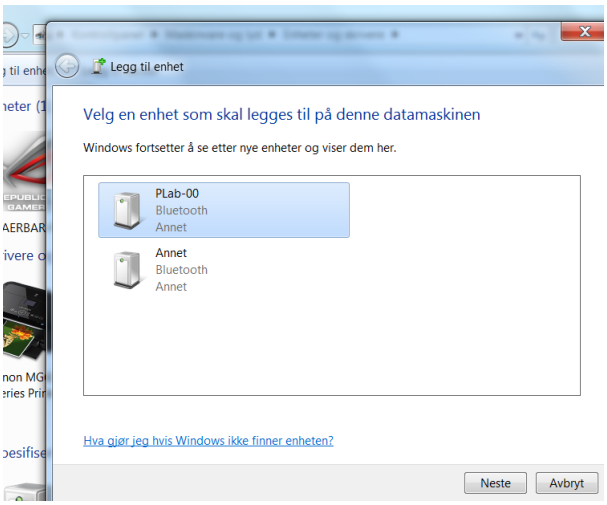
Processing på datamaskina og bluetooth

For å kommunisere med bluetooth, må maskina ha støtte for bluetooth og du må først pare enheten med maskina di. I Windows 7 kan du legge til nye bluetooth enheter når bluetooth radioen står på ved å gå til Start->Enheter og skrivere. Velg "Legg til enhet". Eneheten du skal pare skal nå komme opp i lista. Velg denne enheten og trykk på neste. Velg "Angi enhetens paringskode". Standard paringskode er "1234", skriv inn dette i vinduet som kommer opp. Enheten blir så installert, og får tildelt en com port. Gjerne merk deg denne. Du kan få tak i den når som helst, beskrevet under. Steg for steg ser tilkoblingsprosedyren slik ut:

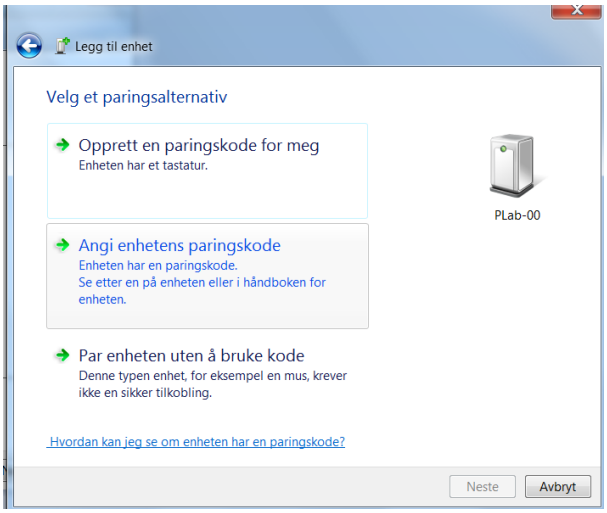
Start menu: 

Velg "Legg til enhet": 

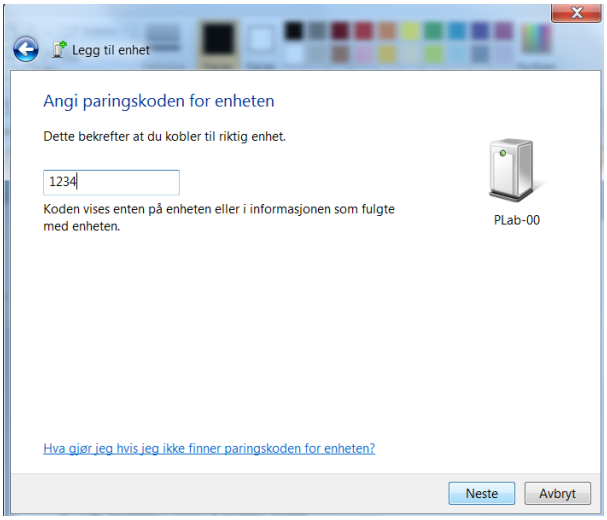
Velg enheten:



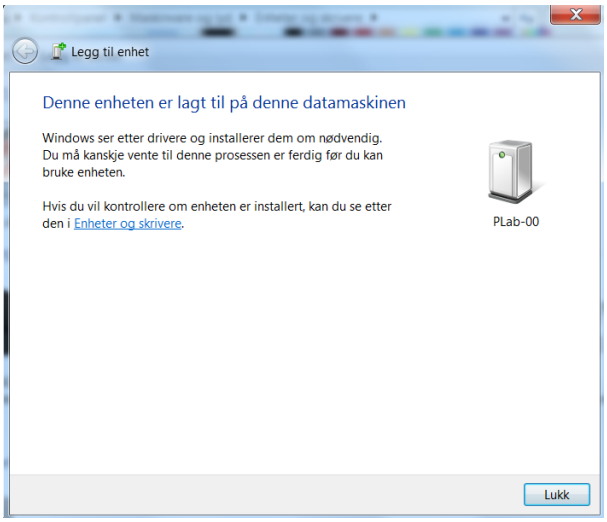
Velg "Angi enhetens paringskode":



Skriv inn koden og trykk "Neste":

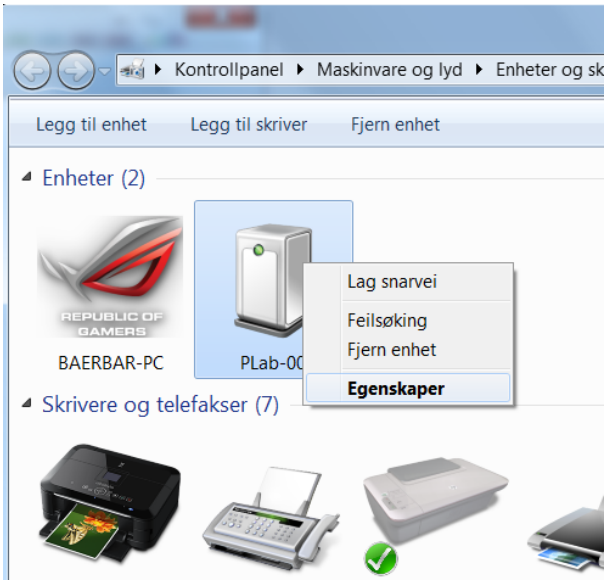


Du er nå ferdig. Trykk "Lukk":

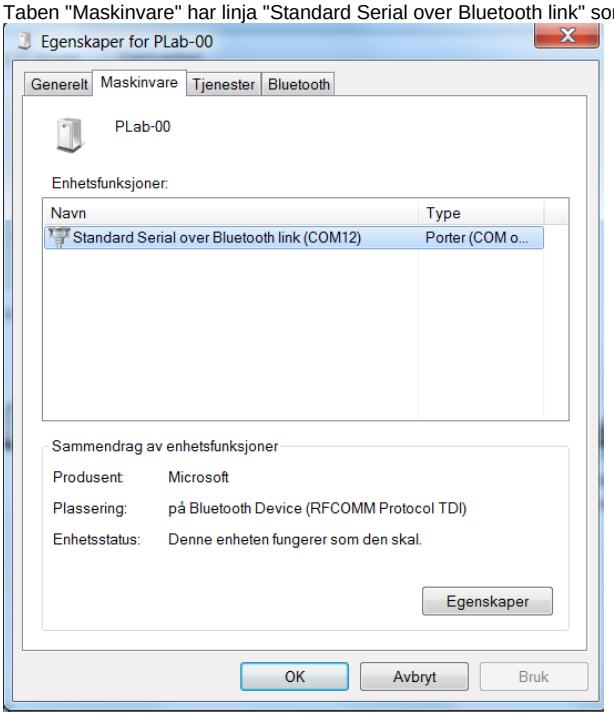


Om du mister det kan du alltid finne den tilbake ved å gå tilbake til "enheter og skrivere", se på egenskapene dens under taben "Maskinvare". Com port nummeret skal vises på linja "Standard Serial over Bluetooth Link (COMXX)" hvor COMXX er com porten den bruker.

Velg egenskaper for enheten:



Taben "Maskinvare" har linja "Standard Serial over Bluetooth link" som inneholder navnet på porten:



Tilkobling fra kode, bruk AppReplacer

Etter paring på mac er det litt enklere å koble til enn på Windows, da enheten får navn etter hva den faktisk heter. Standard for dere er PLabXX (XX er her gruppenummer). Vi har lagd kode som dere kan bare klippe ut og lime inn som skal lete etter den første enheten som er paret som starter med PLab, og velge denne.

Når du har paret en enhet kan du bruke datamaskina som kontroll på samme måte som du kan bruke appen vår. Vi har lagd kopier-og-lim-inn kode som skal hjelpe deg med det, og gjøre det rett fram å overføre koden du har skrevet til mobil enhet. Koden ligger som del av ArduinoMobileIntegrationExamples/MinimalBTExample/Processing/MinimalBTExample, og er fila AppReplacer. Bare kopier denne fila til prosjektet ditt, endre navn på kom port/ fjern kommentering på søkealgoritmen og kall setupSerial() fra den vanlige setup() metoden for å få det hele til å virke.

For at det skal virke, må dere kalle setupSerial() fra setup(). Ellers er det bare å kjøre vanlig Processing utvikling og kode.