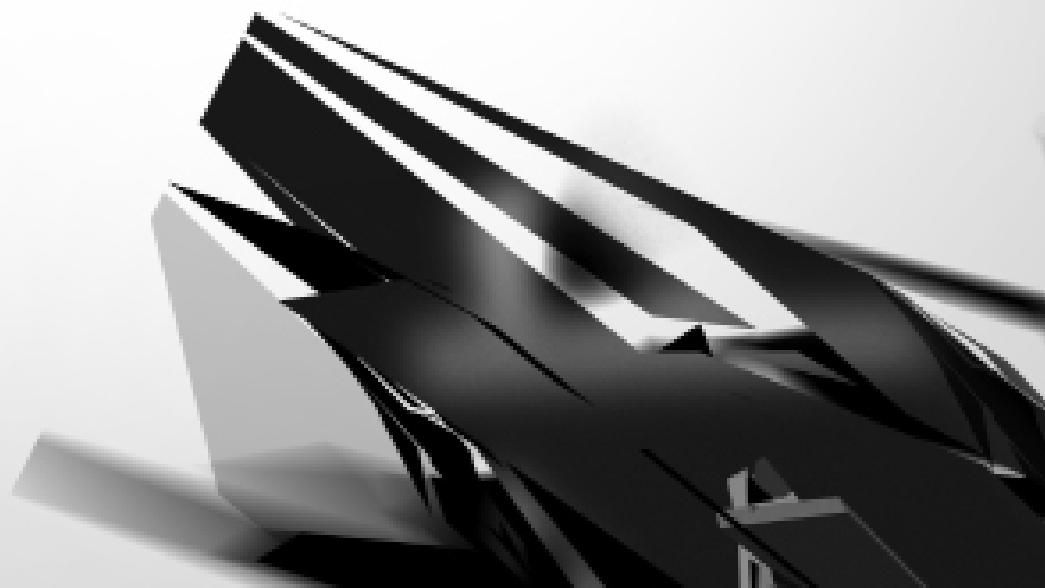


Honigstein Samuel

CAHIER DE RÉALISATION

CONCEPTEUR RÉALISATEUR MULTIMÉDIA



SOMMAIRE

PROJET TUTEURÉ, PROJET PROFESSIONNEL

DISPOSITIF NUMÉRIQUE INTERACTIF TEMPS RÉEL : LIGHTIZON

3

PROJET PERSONNEL

PORTFOLIO GÉNÉRATIF

7

PROJET CONCOURS : BATTLE GRAPHIQUE

VISUEL INTERACTIF GÉNÉRATIF : ISEA

8

PROJET SCOLAIRE

JEU VIDÉO, JEU D'AVVENTURE RPG : ZELDA

9

JEU VIDÉO, JEU DE PLATEFORMES : MEGAMAN

10

APPLICATION MOBILE ANDROID : CHAT, MESSAGERIE

11

JEU PHP : LA TOUR DE PISE

12

CARNET DE VOYAGE EN XML

13

PROJET TUTEURÉ, PROJET PROFESSIONNEL

DISPOSITIF NUMÉRIQUE INTERACTIF TEMPS RÉEL : LIGHTIZON



LightizOn est un projet d'installation numérique interactive de light painting video en temps réel dans lequel j'interviens en qualité de chef de projet, développeur interactif. Je suis responsable de la gestion du projet et du développement logiciel de l'installation numérique.

Le programme Lightizon capture un flux vidéo puis traite les images au fur et à mesure qu'elles sont rendues disponibles. Un algorithme distingue les pixels plus lumineux à travers un moteur de rendu et ré-affiche à l'écran les traînées lumineuses en temps réel.



Les trois captures ci-dessus illustrent le déroulement du rendu. La photo évolue au fur et à mesure que l'on agite des lumières devant le capteur, le but étant de créer des peintures numériques.

PLUSIEURS VERSIONS DU PROGRAMME ONT ÉTÉ DÉVELOPPÉ :

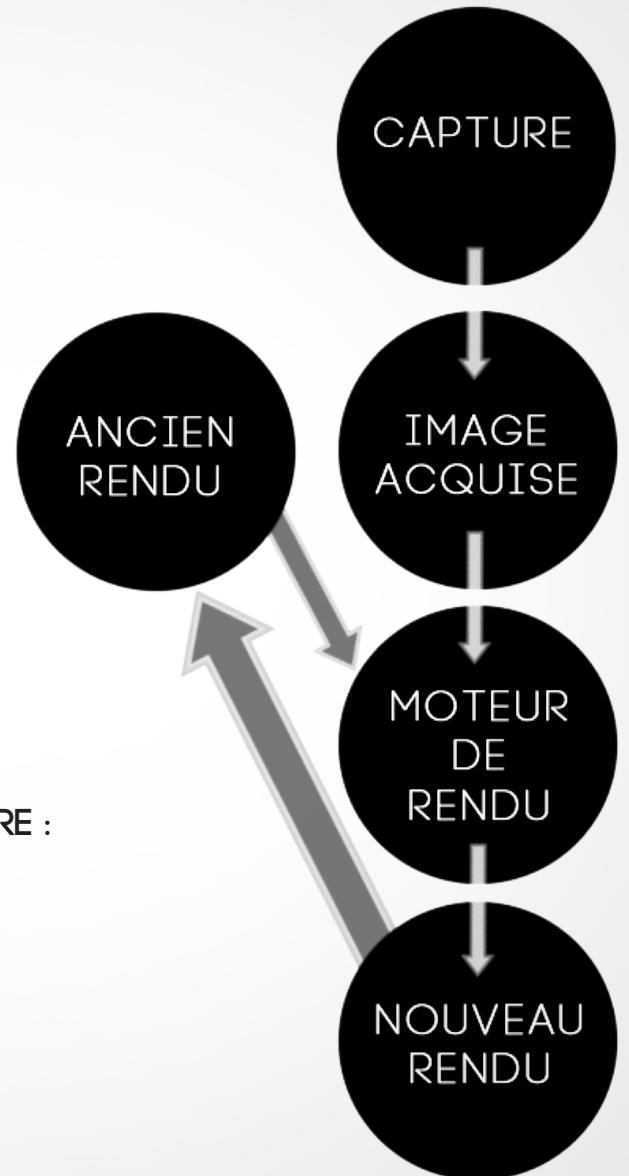
La version Processing a servi à prototyper rapidement l'algorithme du programme. Cette version inclut également un serveur Syphon (protocole existant uniquement sur OSX) permettant de créer un tube vidéo en le sketch Processing et les différents logiciels de mapping vidéo. Un module de capture vidéo qui convertit l'avancement du sketch dans un fichier vidéo .AVI a également été monté sur cette version (librairie GSVIDEO).

La version Java sous Eclipse implémente le framework de Canon, et a permis non seulement d' accéder au buffer d'un appareil numérique Canon, et de l'utiliser comme acquisiteur d'image, mais également de profiter de tout l'hardware de l'appareil photo numérique en terme de réglage : iso, ouverture focale, vitesse d'obturation, balance des blancs.

Une version OpenFramework (C++) a été mis en place pour comparer les temps de calcul de l'exécution Java et C++.

PARMI TOUTES CES VERSIONS, LE SCHÉMA D'EXÉCUTION DU PROGRAMME EST SIMILAIRE :

Un objet capture récupère l'image provenant du capteur lorsque celle-ci est rendue disponible. L'image acquise entre dans le moteur de rendu pour être comparé à l'ancien rendu. Un algorithme de comparaison d'image détecte les pixels lumineux et les affiche dans le nouveau rendu. Pour finir, le nouveau rendu devient un ancien rendu.



LightPaintingThread | Processing 2.0b6

File Edit Sketch Tools Help

LightPaintingThread control engine reverse

```

void light() {
    // threadid correspond à l'ID du thread qui a été lancé
    // A partir de cet ID, les différents threads se divisent le parcours de l'image
    int threadid=coreID;
    flagon[threadid]=true;
    // r g b, sont les valeurs entières correspondant aux couleur rouge, vert et bleu.
    int r, g, b;
    // le parcours de l'image est divisé.
    // Chaque thread s'occupe d'une partie distincte de l'image
    // ce qui permet d'optimiser considérablement le temps de calcul.
    for (int id=threadid*(pix/corenb);id<(threadid+1)*(pix/corenb);id++) {
        // c récupère la couleur du pixel en cours en format 32 bits.
        c=cam.pixels[id];
        // à l'aide d'un right shift, le format 32 bits est découpé en format 8 bits.
        // récupérant ainsi une valeur de couleur comprise entre 0 et 255
        r=c>>16&0xFF;
        g=c>>8&0xFF;
        b=c&0xFF;
        // c récupère l'ancienne image de rendu, pour pouvoir en comparer les pixels,
        // et déterminer si le nouveau pixel est plus lumineux que l'ancien.
        c=pixels[id];

        // La comparaison entre la nouvelle image, et le dernier rendu sont effectués
        // pour déterminer si le pixel est plus lumineux que l'ancien
        // à travers une opération ternaire
        //fr = 0.9999 et correspond à la persistance de la vision du pixel.
        // plus cette valeur tend vers 1, plus le pixel lumineux persistera à l'écran.
        // a l'inverse, si cette valeur est basse, alors le pixel lumineux disparaîtra avec le temps
        r=(int) ((fr*(c>>16&0xFF))>>16);
        g=(int) ((fr*(c>>8&0xFF))>>8);
        b=(int) ((fr*(c&0xFF))>>0);

        // réassemblage de la couleur ( left shift )
        r=(r<<16);
        g=(g<<8);
        // nouveau rendu de l'image
        pixels[id]=a|r|g|b;
    }
    // flag permettant d'indiquer à la fonction principale que le thread a fini son calcul
    flagoff[threadid]=true;
}

```

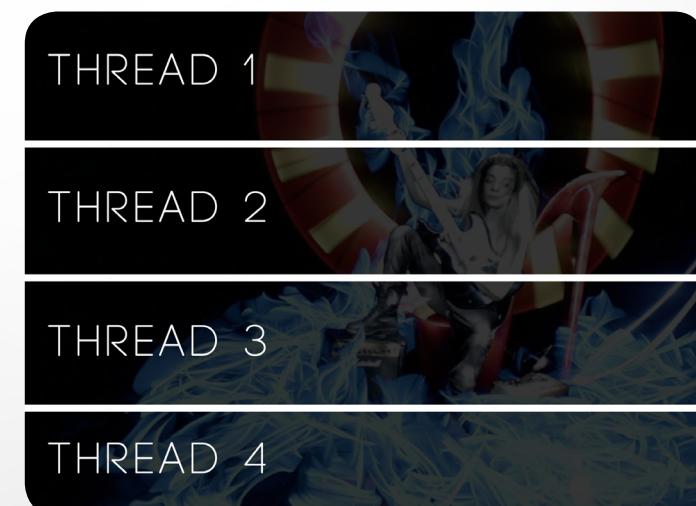
Voici le moteur de rendu correspondant à la version Processing de LightizOn. Cette version fonctionne exclusivement avec des webcams. La fonction Light() désassemble la couleur 32 bits, en 4 entités distinctes de 8 bits : alpha, rouge, vert , bleu, avec l'aide des «right shift». Ceci permet de manipuler la couleur dans un rang de 0 à 255 et comparer chaque pixel avec celui de l'image précédente. La couleur est ensuite réassemblée pour être affichée : «left shift».

32 BITS



8 BITS 8 BITS 8 BITS 8 BITS

L'algorithme général est monté en multithreading pour permettre une vitesse de calcul approchant le temps réel. En divisant le parcours, chaque thread s'occupe d'une partie distincte de l'image.



La version Java, développée sur Eclipse est construite sur le schéma de conception Observateur / Observable. La vue observe le modèle. Cette version plus évoluée permet l'acquisition d'image depuis tous les appareils photos numériques de la gamme Canon. Le moteur de rendu a été implémenté dans le Framework. Voici deux extraits de scripts, à gauche, le modèle qui hérite de la classe Observable, et à droite la vue qui implémente Observer. La vue est notifié lorsque celle-ci doit afficher l'image à l'écran.

OBSERVABLE

```
CanonMain.java *CanonMDL.java *CanonView.java

BufferedImage light = new BufferedImage(1056, 704, BufferedImage.TYPE_INT_ARGB);
BufferedImage image = new BufferedImage(1056, 704, BufferedImage.TYPE_INT_BGR);

// constructeur du modèle
public CanonMDL() {
    // initialisation de la caméra Canon
    final CanonCamera cam = new CanonCamera();
    // wr est un WritableRaster qui permet d'écrire très rapidement dans des BufferImage
    wr = light.getRaster();
    // initialisation de la vue
    CanonView view= new CanonView(cam,this);
    // ouverture de la session Canon, et lancement de la LiveView de Canon.
    cam.openSession();
    cam.beginLiveView();
    // boucle infinie
    while( true ){
        // récupération de l'image de Canon
        BufferedImage image = cam.downloadLiveView();
        if( image != null ){
            //System.out.println(image);
        // lancement d'un thread qui correspond au moteur de rendu
            Thread buffrgb = new buffrgb(image);
            buffrgb.start();
        // vidage du BufferImage Image
            image.flush();
        }   }
}

// la fonction observ est lancée à la fin du traitement du moteur de rendu
// et permet de notifier la vue ( modèle Observateur, Observable ).
// le modèle extends Observable
public void observ(){
    this.setChanged();
    this.notifyObservers();
}
```

OBSERVEUR

```
CanonMain.java CanonMDL.java *CanonView.java

// cette classe implements Observer
// et observe la classe CanonMDL ( modèle de conception Observateur/ Observable )
public class CanonView implements Observer, ActionListener {
    CanonCamera cam;
    CanonMDL modele;    JFrame frame;    JLabel label;

    public CanonView(final CanonCamera cam, CanonMDL mdl){
        // récupération du modèle
        modele =mdl;
        // ajout d'un écouteur, la vue écoute désormais le modèle.
        modele.addObserver(this);
        frame = new JFrame( "LightizOn" );
        frame.addKeyListener(new Controleur());
        // setUndecorated, sert à supprimer les bordures d'une fenêtre.
        frame.removeNotify();
        frame.setUndecorated(true);
        frame.addNotify();
        label = new JLabel();
        frame.getContentPane().add( label, BorderLayout.CENTER );
        frame.setDefaultCloseOperation( JFrame.DO_NOTHING_ON_CLOSE );
        frame.addWindowListener( new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                cam.endLiveView();
                cam.closeSession();
                CanonCamera.close();
                System.exit( 0 );
            }
        });
        frame.setVisible( true );
    }
    // la fonction update est lancée lorsque le modèle lance un notifyObservers();
    public void update(Observable arg0, Object arg1) {
        // le BufferedImage "light" du modèle est affiché à l'écran.
        label.setIcon( new ImageIcon( modele.light ) );
        frame.pack();
    }
}
```

Pour finir, le dispositif interactif LightizOn a eu la chance de se produire dans différents haut lieux parisiens comme la Cité de la Mode et du Design, l'espace Pierre Cardin, la Machine du Moulin Rouge, et le musée du Quai Branly. D'autres interventions sont prévus. Ce projet a gravit les échelons passant du statut de projet étudiant, à celui de projet professionnel.

PROJET PERSONNEL

PORTOFOLIO GÉNÉRATIF



The screenshot shows a generative portfolio website. At the top left is a greeting 'HELLO SAM.'. Below it is a navigation bar with links: 'GRAPHIC DESIGN / VIDEO GAMES / MEDIA EVENTS / ABOUT ME'. The main content area has a title 'WELCOME TO MY UNDERWORLD' and a list of 'OPEN FRAMEWORKS' including: PROCESSING, ACTIONSCRIPT, JAVASCRIPT, XML PHP, JAVA C++, ANDROID, IOS, J2ME, INDESIGN, ILLUSTRATOR, PHOTOSHOP, and EDGE ANIMATE.

Voici trois extraits de scripts, le premier script est un Font-Kit qui permet d'utiliser et d'afficher la police CodeLight sur tous les navigateurs. Le second, est un extrait de JQuery, qui lance différentes fonctions lorsque l'on clique sur l'onglet «Events». Le dernier script un extrait de MediaQuery qui repositionne et redimensionne les différents éléments en fonction de la taille de l'écran.

```
font-face {
  font-family: 'Codelight';
  src: url('font/code_light-webfont.eot');
  src: local('@'), url('font/code_light-webfont.woff') format('woff'),
  url('font/code_light-webfont.ttf') format('truetype'),
  url('font/code_light-webfont.svg') format('svg');
  font-weight: normal;
  font-style: normal;
```

```
// lorsque l'on clique sur l'ID event
$("#events").click(function() {
slide=false; // la valeur slide passe à faux
$("#bandeau").stop(); // le bandeau arrête son animation
// puis il se retrécit en une seconde
$("#bandeau").animate({width:'0%'},1000);
// tous les texte sont cachés.
$("#txt").hide(700);
// la valeur son passe à faux
sound=false;})
```

```
/* si la largeur de l'écran est en dessous de 800 px */
@media only screen and (max-width:800px) {
.txt{font-size:20px; /* la taille du texte se retrécit*/
width:150px;
}
.txtarticle{font-size:18px;
}
/* ta taille du bandeau World se retrécit et se replace*/
#world{background-size: 80% 80%;
right:20%;
top:220px;
margin:auto;
}/* la taille du bandeau menu se retrécit*/
#bandeau{height:60px;}
/* la taille du bandeau games se retrécit et s replace*/
#games{background-size: 100% 100%;
top:170px;
}
```

PROJET CONCOURS : BATTLE GRAPHIQUE

VISUEL INTERACTIF GÉNÉRATIF : ISEA



Le projet Isea est un rendu produit sur 8 heures, issu d'une battle graphique entre l'IUT de l'université Paris XIII et celui de Sarcelles. Isea est une conférence internationale qui se déroule à Kyoto tous les ans rapprochant la nature et la technologie. Pour Isea, nous devions créer un logo (la petite feuille), une accroche (La technologie imite sa nature), ainsi qu'un visuel (la photo Light Painting est une de mes photos). Sur ce projet, je suis allé plus loin que le simple projet de graphisme en produisant un format interactif pour le web. La feuille à droite de l'écran se plie et se déploie en fonction de la position (X, Y) de la souris à l'aide de la technologie Processing retraduit en Canvas. Processing permet de produire très rapidement du contenu interactif avec des lignes très fines, tout en restant élégant. La fonction récursive branch permet de dessiner les fractales du visuel en y introduisant les positions X et Y de la souris.



```
P BattleIsea | Processing 2.0b6
File Edit Sketch Tools Help
BattleIsea §
void branch(float len,int num)
{
    // a chaque lancement de la fonction, len est décrémenté
    // en le multipliant par le coefficient f
    // la valeur len ou lenght, est utilisé pour faire varier la taille
    // de chaque branche au fur et à mesure que la fonction est relancée
    // len est calculé de base en fonction de la hauteur du sketch.
    len *= f;
    // num permet de contrôler le nombre de fois que cette fonction récursive va tourner sur elle même
    num -= 1;
    // si len est au dessus de 1,
    if((len > 1) && (num > 0))
    {
        //pushMatrix sauvegarde l'actuel avancement du sketch
        pushMatrix();
        // rotation de curlx, curlx est calculé en fonction de la position X de la souris sur le sketch
        rotate(curlx);
        // dessin d'une ligne de taille -len
        line(0,0,0,-len);
        // translation de -len
        translate(0,-len);
        // lancement de la fonction branch avec d'autres paramètres.
        branch(len,num);
        // restauration du sketch avec son nouvel avancement
        popMatrix();
        len *= growth;
        pushMatrix();
        // cette rotation permet de dérouler la branche dans la partie inférieur
        // lorsque curly est supérieur à curlX
        rotate(curlx-curly);
        // dessin de la ligne
        line(0,0,0,-len);
        // translation
        translate(0,-len);
        // lancement de la fonction branch avec d'autres paramètres
        branch(len,num);
        // restauration des coordonnées du sketch.
        popMatrix();
    }
}
Done Saving.
56
```

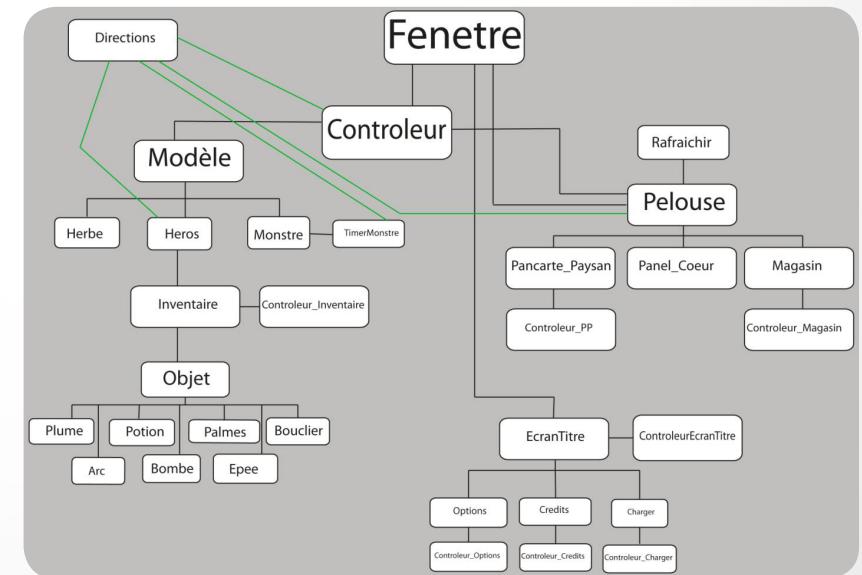
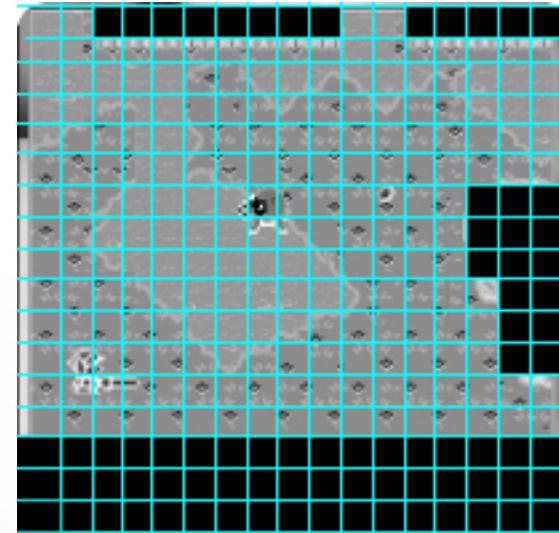
PROJET SCOLAIRE

JEU VIDÉO, JEU D'AVVENTURE RPG : ZELDA



Zelda est un remake d'une fabuleuse saga de jeu vidéo RPG (Role Playing Game). Il s'agit d'un travail scolaire réalisé en binôme. Il était question lors du développement de ce jeu vidéo en JAVA, de réaliser en amont un cahier des charges, et un cahier technique, évoquant le modèle de conception du jeu. Cet exercice a permis d'évaluer notre niveau en développement JAVA, en programmation orienté-objet, notre maîtrise du modèle de conception MVC mais également notre capacité à prévoir et mettre en place des solutions techniques.

Zelda inclut dans son développement, un système d'attaque des ennemis avec plusieurs armes, le ramassage des objets, une gestion de l'inventaire, une gestion de la vie du héros et des ennemis, une matrice de restriction de déplacement du personnage, un magasin d'objet, de la micro-intelligence artificielle des ennemis.



Parmi les différentes solutions techniques finalement mis en place, une d'entre-elle gère la restriction de déplacement du personnage sur des zones non accessibles. Le terrain est en réalité une énorme matrice sur lequel des valeurs booléennes indiquent si le terrain est praticable ou pas (les zones en noir ne sont pas accessibles par le personnage.)

Ci-dessus, un schéma de l'allure principale du jeu, divisé par classe, tel qu'il a été prévu dans le cahier technique du jeu.

PROJET SCOLAIRE

JEU VIDÉO, JEU DE PLATESFORMES : MEGAMAN



Megaman est le remake d'un jeu de plate-formes qui est apparu sur la console NES dans les années 1990. Il s'agit d'un travail scolaire pour juger nos capacités de développement en ActionScript 3.0, mais également de notre niveau en programmation orienté-objet. Le jeu a été développé en modèle de conception MVC (modèle / vue / contrôleur).

Megaman inclut dans son développement, de l'animation en ActionScript 3.0, du scrolling horizontal (fond et plates-formes) lors du déplacement du personnage, la détection des collisions du personnage avec les plates-formes, la détection des collisions des tirs avec les ennemis, le ramassage des objets, la gestion du score, la gestion de vie et de mort, de la micro-intelligence artificielle des ennemis, ainsi qu'une gestion de la gravité du personnage lors des sauts.

```
// la fonction recontresol teste si une plateforme est touché par le personnage
public function rencontreSol()
{
    // initialisation d'un sprite à null
    // celui ci est renvoyé à nul lorsque rien n'est touché
    var sp:Sprite;
    sp = null;
    // parcours du tableau de plateforme du paysage
    for (var i=0; i < monpays.tabplat.length; i++)
    {
        // si le personnage touche la plateforme,
        // et que la valeur posX est le négatif de la valeur x de la plateforme
        // et que la valeur y du personnage est différente
        // a 40 pixel près de la valeur posy de la plateforme
        if (this.hitTestObject(monpays.tabplat[i]) && posX==monpays.tabplat[i].x
            && (this.y-40<=monpays.tabplat[i].posy))
        {
            // alors la plateforme est touché
            sp = monpays.tabplat[i];
        }
    }
    // et est renvoyé en retour de la fonction
    return sp;
}
```



L'extrait de script test si le personnage touche un sprite ou non. Si le personnage touche un sprite, le sprite est retourné, la valeur Y du sprite est alors utilisé pour placer le personnage sur la plate-forme. Sinon, le sprite est renvoyé à «null».

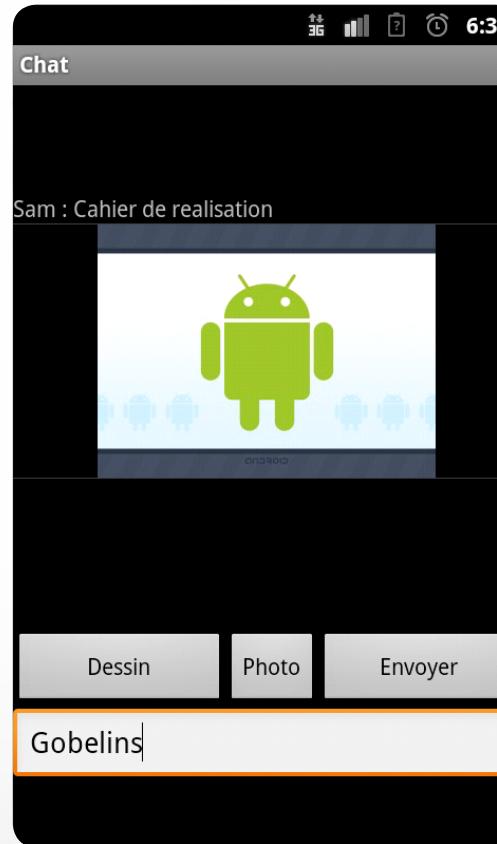
APPLICATION MOBILE ANDROID : CHAT, MESSAGERIE



Le chat Android, est un projet scolaire réalisé dans le cadre de l'option multimédia orienté mobile. Il s'agit d'un chat développé en client / serveur permettant d'envoyer des messages, des photos et du dessin. L'extrait de script détail la façon dont le client transforme le message, la photo, le dessin, en tableau de bytes, de façon à l'envoyer vers le serveur. En fonction du premier Byte que le serveur reçoit, celui-ci sait si il reçoit une image (I), ou bien un texte (T).

```

public void run() {
    // si le bitmap est null
    if (bitmap == null) {
        int bytes = msg.length(); // on récupère la taille du message
        // Transformation du message en tableau de 4 bytes
        byte[] minInt = ByteBuffer.allocate(4).putInt(bytes).array();
        try {
            // on envoie "T" dans le flux, qui sera interprété par le serveur
            // par la réception d'un texte
            ds.writeByte('T');
            ds.write(minInt); // écriture du tableau de Bytes dans le flux
            // le message est envoyé
            ds.write(msg.getBytes());
            ds.flush();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
    // sinon c'est qu'il s'agit d'un BITMAP
    else {
        int bytes;
        // transformation du BITMAP, en PNG, puis en tableau de BYTES
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        // compression du bitmap en PNG
        bitmap.compress(Bitmap.CompressFormat.PNG, 100, stream);
        byte[] array = stream.toByteArray(); // transformation du PNG en tableau de Bytes
        bytes = array.length;
        // transformation du tableau de bytes en tableau de 4 bytes
        byte[] minInt = ByteBuffer.allocate(4).putInt(bytes).array();
        try {
            // l'on écrit I dans le flux, indiquant au serveur que l'on envoie une IMAGE
            ds.writeByte('I');
            ds.write(minInt);
            ds.write(array); // et envoi du tableau de bytes en question.
            ds.flush();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
}}
```



PROJET SCOLAIRE

JEU PHP : LA TOUR DE PISE

La tour de pise est en jeu en PHP. Le but du jeu étant de remplacer les briques de sa tour, par celles proposer aléatoirement afin d'avoir une tour aux nombres croissant. Ce jeu inclut le fait que le joueur doit se «logger» pour pouvoir jouer.

Un script php est prévu pour tester si le login et le mot de passe entré dans les champs existe dans la base de donnée prévu. Auquel cas, le joueur peut se créer un login et un mot de passe.

Voici un extrait de script le classe requête du jeu. Ce script teste si le login et le mot de passe existe dans la base de donnée. Si les identifiants existent alors le jeu est lancé par l'envoie du paramètre «game» dans l'url de l'index.php. Si les identifiants n'existent pas, un message indiquant que les identifiants sont incorrect apparaît à l'écran.

The screenshot shows the 'TOUR DE PISE LE JEU' application. On the left, there's a 'CONNEXION' section with 'LOGIN' and 'MDP' input fields. In the center, there's an 'INSCRIPTION' section with similar 'LOGIN' and 'MDP' input fields. Below these, a message says 'VOS IDENTIFIANTS SONT INCORRECTS.' On the right, there are two columns: 'ORDINATEUR' and 'JOUEUR'. Each column has a vertical stack of numbered boxes (16, 37, 36, 44, 49 for O; 14, 15, 13, 33, 8 for J). Between them are two small buttons: '18' and '?'. At the bottom right of the main area, there's a 'DECONNEXION' link.



```
class Requete
{
    var $db = "a7552264_pise"; // nom de la BD
    var $host = "mysql4.000webhost.com"; // nom de la machine hôte
    var $user = "a7552264_user"; // nom de l'utilisateur
    var $pwd = "xxxxxxxx"; // mot de passe
    var $cnxDB;
    var $msg;
    var $passVerif;

    function __construct()
    {
        $dsn = "mysql:host=".$this->host.";dbname=".$this->db; // récupération des datas
        $this->cnxDB = new PDO($dsn, $this->user, $this->pwd); // création d'un PDO
    }

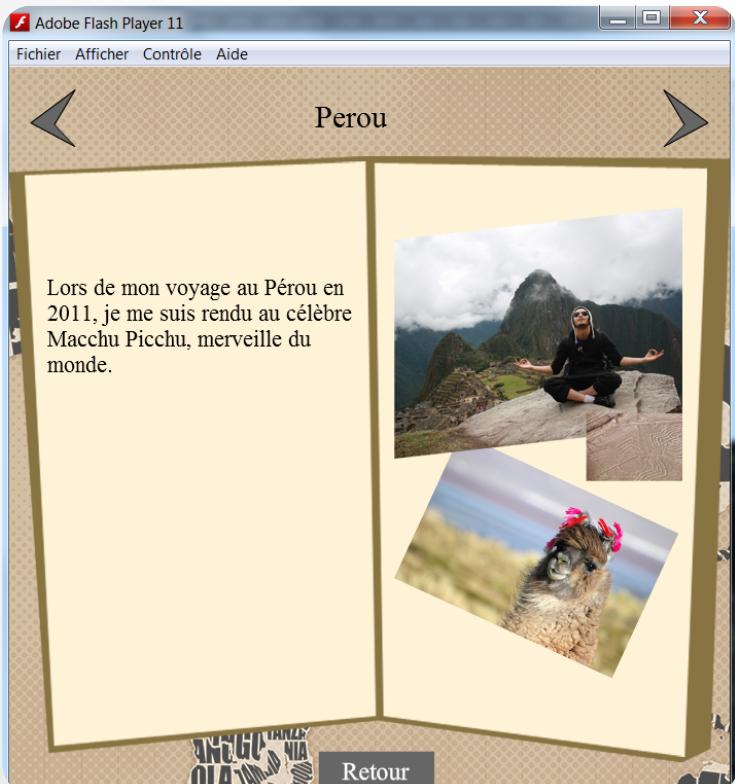
    function connexion($login,$pass)
    {
        // si un des deux champs ne sont pas remplis
        if($login =="" || $pass=="")
        {
            $this->msg="Tous les champs ne sont pas remplis";
        }
        // requete sql, recherche du login
        $sql = 'SELECT mdp FROM user WHERE login="'.$login.'"';
        $resultat = $this->cnxDB->query($sql);
        $passVerif="";
        // recherche du mdp du login dans la base de donnée
        foreach ($resultat as $row)
        {
            $passVerif=$row['mdp'];
        }
        // comparaison du mot de passe de la BDD, et du mdp donné par l'user
        if($passVerif==$pass)
        {
            // variable de session connecté passe à vrai,
            // la variable de session login récupère le nom du logger.
            $_SESSION["connecte"] = true;
            $_SESSION["login"] = $login;
            ?>
            <script language="JavaScript">
                // lancement de l'index.php avec pour paramètre la variable "game"
                document.location.href="index.php?p=game";
            </script>
            <?php
        } else { // sinon les identifiants sont incorrects.
            $this->msg=("Vos identifiants sont incorrects.");
        }
    }
}
```

PROJET SCOLAIRE

CARNET DE VOYAGE EN XML



Le carnet de voyage est un exercice scolaire dans lequel nous devions utiliser une liste XML pour charger les différents éléments du carnet, comme les images, le texte, et le son. L'extrait de script découpe le fichier XML dans une liste d'éléments. Un événement dispatch est envoyé pour notifier la fonction main du bon chargement du fichier XML.



```
// class Chargeur XML permet de charger un fichier XML en mémoire
public class ChargeurXML extends EventDispatcher {
    // URL request pour l'url du fichier XML en question puis chargeur
    var fichierXML:URLRequest;
    var chargementXML:URLLoader;
    // donnée et liste, pour associer a un tableau d'objet le fichier XML
    var Donnees:XML;
    var liste:XMLEList;

    // constructeur du chargeur de XML
    public function ChargeurXML()
    {
        Donnees = null;
        liste = null;
        // Nouveau fichier XML dans fichierXML
        fichierXML = new URLRequest("FichierXML.xml");
        // initialisation du chargeur de XML
        chargementXML = new URLLoader();
        // la suite du code continue que lorsque la fonction traiter XML est terminée
        chargementXML.addEventListener(Event.COMPLETE, traiterXML);
        // chargement du fichier dans le chargeurXML;
        chargementXML.load(fichierXML);
    }

    // la fonction traiter XML découpe le fichier XML dans un objet XML et XML list
    // permet de recuperer les données du fichier xml en question
    public function traiterXML(e:Event)
    {
        Donnees = new XML(e.target.data);
        liste = Donnees.elements();
        // ajout d'un dispatch event permettant a la fonction principale de lancer la fonction charger liste
        // ceci permet de synchroniser le chargement du fichier XML, la récupération de donnée de la listeXML
        // et la classe Main
        dispatchEvent(new Event("XMLcharge"));
    }
}
```