## 代码来自qml葵花宝典一书

## 1.简单使用

项目结构:

|  |
| --- |
|  |

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  Canvas{  id:*canvas*  width: 300  height: 400  onPaint: {  let *ctx* = *getContext*("2d")  *ctx*.lineWidth = 5  *ctx*.strokeStyle = "deeppink" //边框颜色  *ctx*.fillStyle = "yellow"  *ctx*.beginPath()  *ctx*.moveTo(50,50)  *ctx*.lineTo(150,50)  *ctx*.lineTo(80,150)  *ctx*.lineTo(50,50)  *ctx*.closePath()  *ctx*.stroke()  *ctx*.fill()  }  } |  |

## 2.方便API

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  Canvas{  id:*canvas*  width: 300  height: 400  onPaint: {  let *ctx* = *getContext*("2d")  *ctx*.lineWidth = 5  *ctx*.strokeStyle = "deeppink" //边框颜色  *ctx*.fillStyle = "yellow"  //方便调用的API  *ctx*.fillRect(30,30,90,90)  *ctx*.clearRect(40,40,70,70) //可以起到切割矩形的效果  *ctx*.strokeRect(20,20,65,65)  }  } |  |

## 3.阴影效果

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  Canvas{  id:*canvas*  width: 300  height: 400  onPaint: {  let *ctx* = *getContext*("2d")  *ctx*.lineWidth = 5  *ctx*.strokeStyle = "deeppink" //边框颜色  *ctx*.fillStyle = "yellow"  //shadow属性  *ctx*.shadowColor="green"  *ctx*.shodowOffsetX =3  *ctx*.shadowOffsetY =3  //ctx.strokeRect(20,20,65,65)  //ctx.fillRect(10,10,50,50)  *ctx*.font="bold 80px Arial" //设置字体字号  *ctx*.fillText("hi,girls",30,180)  }  } |  |

## 4.绘制和裁剪图片,这个功能需要使用QquickView来加载qml文件

|  |  |
| --- | --- |
| //main.cpp  #include <QGuiApplication>  #include <QQmlApplicationEngine>  #include<QQuickView>  int **main**(int **argc**, char \***argv**[])  {  #if QT\_VERSION < QT\_VERSION\_CHECK(6, 0, 0)  QCoreApplication::setAttribute(Qt::AA\_EnableHighDpiScaling);  #endif  QGuiApplication **app**(*argc*, *argv*);  //使用QQuickView,如果需要使用Canvas来绘制图像需要这种方式  QQuickView\* **view** = new QQuickView;  const QUrl **url**(QStringLiteral("qrc:/main\_nowin.qml"));  view->setSource(url);  view->show();  return app.exec();  } | //main\_nowin.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Canvas{  id:*root*  width: 400  height: 300  property string imgPath:"./ball.png"  // property string imgPath: "./scene2.png"  //property string imgPath: "./songshu.jpg"  onPaint: {  var *ctx* = *getContext*("2d")  *ctx*.strokeStyle='#ff2a68'  //draw the image  *ctx*.drawImage(*imgPath*,10,10)  *ctx*.save()  *ctx*.beginPath()  *ctx*.moveTo(110,10)  *ctx*.lineTo(155,10)  *ctx*.lineTo(135,55)  *ctx*.closePath()  // translate coordinate system  *ctx*.clip() // create clip from the path  // draw image with clip applied  *ctx*.drawImage(*imgPath*, 100, 10)  // draw stroke around path  *ctx*.stroke()  // restore previous context  *ctx*.restore()  }  Component.onCompleted: {  *loadImage*(*imgPath*)  }  } |
|  |  |

## 5.变换效果,平移,旋转的效果

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  //变换  Canvas{  id:*cvs*  width: 240  height: 120  onPaint: {  var *ctx* = *getContext*("2d")  *ctx*.strokeStyle = "green"  *ctx*.lineWidth = 4  *ctx*.translate(*cvs*.width/2,*cvs*.height/2)  *ctx*.beginPath()  *ctx*.rect(-20,-20,40,40)  *ctx*.stroke()  *ctx*.strokeStyle="deeppink"  *ctx*.rotate(*Math*.PI/4)  *ctx*.beginPath()  *ctx*.rect(-20,-20,40,40)  *ctx*.stroke()  }  }  } |  |

## 6.组合模式

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  //组合模式（ Composition Mode）  Canvas{  id:*canv*  width: 400  height: 200  onPaint: {  let *ctx* = *getContext*("2d")  //设置组合模式的操作方式为异或  //ctx.globalCompositionOperation = 'xor'  //ctx.globalCompositionOperation = 'and'  *ctx*.globalCompositionOperation = 'or'  *ctx*.fillStyle = "orange"  for(var *i*=0;*i*<40;*i*++){  *ctx*.beginPath()  *ctx*.arc(*Math*.random()\*400,*Math*.random()\*200,20,0,2\**Math*.PI)  *ctx*.closePath()  *ctx*.fill()  }  }  }  } |  |

## 7. 设置组合模式的各种操作方式实例

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Canvas元素")  // 设置组合模式的各种操作方式实例  Canvas {  id: *root*  width: 600; height: 400  property var operation : [  'source-over', 'source-in', 'source-over',  'source-atop', 'destination-over', 'destination-in',  'destination-out', 'destination-atop', 'lighter',  'copy', 'xor', 'qt-clear', 'qt-destination',  'qt-multiply', 'qt-screen', 'qt-overlay', 'qt-darken',  'qt-lighten', 'qt-color-dodge', 'qt-color-burn',  'qt-hard-light', 'qt-soft-light', 'qt-difference',  'qt-exclusion'  ]  onPaint: {  var *ctx* = *getContext*('2d')  for(var *i*=0; *i*<*operation*.length; *i*++) {  var *dx* = *Math*.floor(*i*%6)\*100  var *dy* = *Math*.floor(*i*/6)\*100  *ctx*.save()  *ctx*.fillStyle = '#33a9ff'  *ctx*.fillRect(10+*dx*,10+*dy*,60,60)  // TODO: does not work yet  *ctx*.globalCompositeOperation = *root*.operation[*i*]  *ctx*.fillStyle = '#ff33a9'  *ctx*.globalAlpha = 0.75  *ctx*.beginPath()  *ctx*.arc(60+*dx*, 60+*dy*, 30, 0, 2\**Math*.PI)  *ctx*.closePath()  *ctx*.fill()  *ctx*.restore()  }  }  }  } |  |

## 8.像素缓存功能的实现

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  //像素缓存  Rectangle{  width: 240;height: 120  Canvas{  id: *canvas*  anchors.fill: *parent*  x:10;y:10  property real hue:0.0  onPaint: {  var *ctx* = *getContext*("2d")  var *x* =10+*Math*.random()\*80  var *y* = 10+*Math*.random()\*80  *hue* +=*Math*.random()\*0.1  if(*hue*>1.0) {*hue* -= 1}  *ctx*.globalAlpha = 0.7  *ctx*.fillStyle =*Qt*.hsla(*hue*,0.5,0.5,1.0)  *ctx*.beginPath()  *ctx*.moveTo(*x*+5,*y*)  *ctx*.arc(*x*,*y*,*x*/10,0,360)  *ctx*.closePath()  *ctx*.fill()  }  MouseArea{  anchors.fill: *parent*  onClicked: {  //生成能够被Image使用的url  var *url* = *canvas*.toDataURL("image/png")  *console*.log("url="+*url*)  *image*.source=*url*  }  }  }  Image{  id:*image*  width: 100;height: 100  x:130;y:10  }  Timer{  interval: 1000  running: true  repeat: true  triggeredOnStart: true  onTriggered: {  *canvas*.requestPaint()  }  }  }  } |  |

## 9.Canvas实现简单绘图版功能

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Canvas Demos")  //Canvas实现简单绘图版  Rectangle{  width: 400;height: 300  color: "#333333"  Row{  id:*colorTool*  anchors{  horizontalCenter: parent.horizontalCenter  top: parent.top  topMargin: 8  }  //当前激活的颜色框  property variant activeSquare: *red*  property color paintColor: "#33B5E5"  Repeater{  model:["#33B5E5", "#99CC00", "#FFBB33", "#FF4444"]  ColorSquare{  id: *red*  color: modelData  active: *parent*.paintColor = *color*  onClicked: *parent*.paintColor = *color*  }  }  }  Rectangle{  anchors.fill: *canvas*  border.color: "#666666"  border.width: 4  }  Canvas{  id: *canvas*  anchors{  left: parent.left  right: parent.right  top: *colorTool*.bottom  bottom: parent.bottom  margins: 8  }  //保存上一次的点的坐标  property real lastX  property real lastY  //保存用户选择的颜色  property color color: *colorTool*.paintColor  onPaint: {  var *ctx* = *getContext*("2d")  //设置线宽  *ctx*.lineWidth =1.5  //设置绘制颜色为选中的颜色  *ctx*.strokeStyle = *canvas*.color  *ctx*.beginPath()  *ctx*.moveTo(*lastX*,*lastY*)  //通过MouseArea来获取鼠标点击的点是坐标  *lastX* = *area*.mouseX  *lastY* = *area*.mouseY  *ctx*.lineTo(*lastX*,*lastY*)  *ctx*.stroke()  }  MouseArea{  id:*area*  anchors.fill: *parent*  onPressed: {  *canvas*.lastX = *mouseX*  *canvas*.lastY = *mouseY*  *console*.log(*canvas*.lastX ,*canvas*.lastY)  }  onPositionChanged: {  //引发重绘  *canvas*.requestPaint()  //console.log(mouseX,mouseY)  }  }  }  }  } |  |

## 10 html画布移植,比较复杂需要多查网上资料

### <1>螺旋图形spirograph

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Canvas Demos")  //螺旋图形spiro graph  Canvas {  id: *root*  width: 300; height: 300  // M1>>  onPaint: {  var *ctx* = *getContext*("2d");  *draw*(*ctx*);  }  // <<M1  function *draw*(ctx) {  *ctx*.fillRect(0,0,300,300);  for (var *i*=0;*i*<3;*i*++) {  for (var *j*=0;*j*<3;*j*++) {  *ctx*.save();  *ctx*.strokeStyle = "#9CFF00";  *ctx*.translate(20+*j*\*50,20+*i*\*50);  *drawSpirograph*(*ctx*,20\*(*j*+2)/(*j*+1),-8\*(*i*+3)/(*i*+1),10);  *ctx*.restore();  }  }  }  function *drawSpirograph*(ctx,R,r,O){  var *x1* = *R*-*O*;  var *y1* = 0;  var *i* = 1;  *ctx*.beginPath();  *ctx*.moveTo(*x1*,*y1*);  do {  if (*i*>20000) break;  var *x2* = (*R*+*r*)\**Math*.cos(*i*\**Math*.PI/72) - (*r*+*O*)\**Math*.cos(((*R*+*r*)/*r*)\*(*i*\**Math*.PI/72))  var *y2* = (*R*+*r*)\**Math*.sin(*i*\**Math*.PI/72) - (*r*+*O*)\**Math*.sin(((*R*+*r*)/*r*)\*(*i*\**Math*.PI/72))  *ctx*.lineTo(*x2*,*y2*);  *x1* = *x2*;  *y1* = *y2*;  *i*++;  } while (*x2* != *R*-*O* && *y2* != 0 );  *ctx*.stroke();  }  }  } |  |

### <2>发光线（ Glowing Lines）其实是glowing bezier curve

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Canvas Demos")  //发光线（ Glowing Lines）其实是glowing bezier curve  Canvas{  id:*canvas*  width: 800  height: 450  property real hue : 0  //起点坐标  property real lastX : *canvas*.width\**Math*.random()  property real lastY : *canvas*.height\**Math*.random()  property bool requestLine: false  property bool requestBlank: false  onPaint: {  var *ctx* =*getContext*("2d")  if(*requestLine*){  *line*(*ctx*)  *requestLine* = false;  }  if(*requestBlank*) //这里不能写else if因为在程序启动的时候这两个是同时设置为true的  {  *blank*(*ctx*)  *requestBlank* = false;  }  }  function *line*(ctx){  //保存当前状态  *ctx*.save()  //平移到中心  *ctx*.translate(*width*/2,*height*/2)  //设置缩放系数  *ctx*.scale(0.9,0.9)  //在平移  *ctx*.translate(-*width*/2,-*height*/2)  *ctx*.beginPath()  //设置线宽,随机计算  *ctx*.lineWidth = 4+*Math*.random()\*10  //移动到起点  *ctx*.moveTo(*lastX*,*lastY*)  //重新计算lastX,lastY  *lastX* = *canvas*.width \* *Math*.random()  *lastY*= *canvas*.height \* *Math*.random()  //绘制bezier曲线  *ctx*.bezierCurveTo(*canvas*.width \* *Math*.random(),  *canvas*.height \* *Math*.random(),  *canvas*.width \* *Math*.random(),  *canvas*.height \* *Math*.random(),  *lastX* ,  *lastY* )  //计算hue的值如果大于1,需要减1  *hue* += *Math*.random()\*0.1  if(*hue* >1.0) { *hue* -= 1}  //设置绘制颜色  *ctx*.strokeStyle = *Qt*.hsla(*hue*,0.5,0.5,1.0)  //设置阴影颜色  *ctx*.shadowColor = 'white'  //设置模糊半径  *ctx*.shadowBlur = 10  //绘制  *ctx*.stroke()  //恢复保存的设置  *ctx*.restore()  }  function *blank*(ctx){  *ctx*.fillStyle = *Qt*.rgba(0,0,0,0.1)  *ctx*.fillRect(0,0,*canvas*.width, *canvas*.height)  }  Timer{  id:*lineTimer*  interval: 40  //running: true  triggeredOnStart: true  repeat: true  onTriggered: {  *canvas*.requestLine = true  //记住要重绘  *canvas*.requestPaint()  }  }  Timer{  id:*blankTimer*  interval: 50  //running: true  triggeredOnStart: true  repeat: true  onTriggered: {  *canvas*.requestBlank=true  *canvas*.requestPaint()  }  }  Component.onCompleted: {  *lineTimer*.start()  *blankTimer*.start()  }  }  } |  |