# 1.Glow就是发光效果

## 简单使用实例

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  Rectangle{ //将背景设置为黑色  color: "black"  anchors.fill: *parent*  }  Image{  id:*bfly*  source: "images/hdjlkt.png"  anchors.centerIn: *parent*  sourceSize: *Qt*.size(*parent*.width, *parent*.height)  smooth: true  visible: false //必须设置为隐藏  }  Glow{  anchors.fill: *bfly*  radius: 8  samples: 17  // color: "white"  color: "purple"  source: *bfly*  }  } |  |

## Glow可以填充任何控件，如文本元素

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  Rectangle{ //将背景设置为黑色  //color: "black"  color: "gray"  anchors.fill: *parent*  }  Text {  id: *txt*  text: *qsTr*("Waiting")  font.pixelSize: 100  anchors.centerIn: *parent*  }  Glow{  anchors.fill: *txt*  radius: 8  samples: 17  // color: "white"  color: "yellow"  source: *txt*  }  } |  |

## 还可以设置闪烁动画效果

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  Rectangle{ //将背景设置为黑色  //color: "black"  color: "gray"  anchors.fill: *parent*  }  Text {  id: *txt*  text: *qsTr*("Waiting")  font.pixelSize: 100  anchors.centerIn: *parent*  }  Glow{  id: *glow*  anchors.fill: *txt*  radius: 8  samples: 17  // color: "white"  color: "yellow"  source: *txt*  SequentialAnimation{  loops: Animation.Infinite //只有设置了这个才会一直闪烁  running: true  NumberAnimation{  target: *glow*  property: "spread"  to: 0  duration: 1000  }  NumberAnimation{  target: *glow*  property: "spread"  to: 0.5  duration: 1000  }  }  }  } | 文字闪烁效果会一直持续 |

# 2.Emitter

它属性粒子系统，需要**import QtQuick.Particles 2.15，还需要指定粒子效果的系统**

**实例**

## 粒子图片，最好是透明背景的

|  |  |
| --- | --- |
|  | 或者 |

## 代码

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  import QtQuick.Particles 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  //设置粒子系统  ParticleSystem{  id:*partSys*  }  //Emitter  Emitter{ //发射器  id:*emitter*  anchors.fill: *parent*  width: 300  height: 300  system: *partSys*  emitRate: 100 //发射粒子的数量  lifeSpan: 500 //粒子的生命周期  lifeSpanVariation: 700  size: 15 //发射的粒子的大小  }  //需要发射的粒子图片  ImageParticle{  id:*particle*  source: "images/bluecircle.png"  system: *partSys*  } | //效果： |

## 还可以控制粒子移动方向，如

|  |
| --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  import QtQuick.Particles 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  //设置粒子系统  ParticleSystem{  id:*partSys*  }  //Emitter  Emitter{ //发射器  id:*emitter*  anchors.fill: *parent*  width: 300  height: 300  system: *partSys*  emitRate: 100 //发射粒子的数量  lifeSpan: 500 //粒子的生命周期  lifeSpanVariation: 300//生成间隔时间  size: 15 //发射的粒子的大小  velocity: PointDirection{ //控制粒子生成的方向  // x:-45 //表示往x轴的负方向生成  // x:45 //表示往x轴的正方向生成  // y:-25 //往上生成  y:25 //往下生成  }  }  //需要发射的粒子图片  ImageParticle{  id:*particle*  // source: "images/bluecircle.png"  source: "images/hdjlkt.png"  system: *partSys*  }  } |

## 还可以利用colorVariation属性修改颜色达到更加kool的效果

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  import QtQuick.Particles 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  //设置粒子系统  ParticleSystem{  id:*partSys*  }  //Emitter  Emitter{ //发射器  id:*emitter*  anchors.fill: *parent*  width: 300  height: 300  system: *partSys*  emitRate: 100 //发射粒子的数量  lifeSpan: 500 //粒子的生命周期  lifeSpanVariation: 300//生成间隔时间  size: 15 //发射的粒子的大小  velocity: PointDirection{ //控制粒子生成的方向  // x:-45 //表示往x轴的负方向生成  x:45 //表示往x轴的正方向生成  // y:-25 //往上生成  //y:25 //往下生成  xVariation: 0  yVariation: 100  }  }  //需要发射的粒子图片  ImageParticle{  id:*particle*  // source: "images/bluecircle.png"  source: "images/hdjlkt.png"  system: *partSys*  colorVariation: 0.3 //随机修改颜色  }  } |  |

# 3、Attractor

它可以给粒子系统添加特效，如下面的代码实现鼠标靠近粒子被弹飞的效果

|  |
| --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  import QtQuick.Particles 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  MouseArea{  id: *mar*  anchors.fill: *parent*  hoverEnabled: true  }  //设置粒子系统  ParticleSystem{  id:*partSys*  }  //Emitter  Emitter{ //发射器  id:*emitter*  anchors.fill: *parent*  width: 300  height: 300  system: *partSys*  emitRate: 10 //发射粒子的数量  lifeSpan: 3500 //粒子的生命周期  lifeSpanVariation: 800//生成间隔时间  size: 30//发射的粒子的大小  velocity: PointDirection{ //控制粒子生成的方向  // x:-45 //表示往x轴的负方向生成  x:45 //表示往x轴的正方向生成  // y:-25 //往上生成  //y:25 //往下生成  xVariation: 0  yVariation: 100  }  }  //需要发射的粒子图片  ImageParticle{  id:*particle*  // source: "images/bluecircle.png"  source: "images/hdjlkt.png"  system: *partSys*  colorVariation: 0.3 //随机修改颜色  }  Attractor{  anchors.fill: *parent*  enabled: true  system: *partSys*  pointX: *mar*.mouseX  pointY: *mar*.mouseY  strength: -10000000//该参数为正就是鼠标吸引粒子，为负就是鼠标弹走粒子  affectedParameter: Attractor.Acceleration  //proportionalToDistance: Attractor.Constant //不好  //proportionalToDistance: Attractor.Linear//不好  // proportionalToDistance: Attractor.Quadratic //不好  // proportionalToDistance: Attractor.InverseLinear  proportionalToDistance: Attractor.InverseQuadratic//效果最好  }  } |

**注意：如果将strength修改为正值，则是鼠标把粒子吸引过来**

# 4.页面正在加载的效果

## 将上面的这几个颜色和BusyIndicator组合使用（import QtQuick.Controls 2.15），就可以实现页面正在加载的效果

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtGraphicalEffects 1.12  import QtQuick.Particles 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("qml发光效果")  MouseArea{  id: *mar*  anchors.fill: *parent*  hoverEnabled: true  }  //Glow samples  Rectangle{ //将背景设置为黑色  //color: "black"  color: "gray"  anchors.fill: *parent*  }  Text {  id: *txt*  text: *qsTr*("Waiting")  font.pixelSize: 100  anchors.centerIn: *parent*  }  Glow{  id: *glow*  anchors.fill: *txt*  radius: 8  samples: 17  // color: "white"  color: "yellow"  source: *txt*  SequentialAnimation{  loops: Animation.Infinite //只有设置了这个才会一直闪烁  running: true  NumberAnimation{  target: *glow*  property: "spread"  to: 0  duration: 1000  }  NumberAnimation{  target: *glow*  property: "spread"  to: 0.5  duration: 1000  }  }  }  //设置粒子系统  ParticleSystem{  id:*partSys*  }  //Emitter  Emitter{ //发射器  id:*emitter*  anchors.fill: *parent*  width: 300  height: 300  system: *partSys*  emitRate: 10 //发射粒子的数量  lifeSpan: 3500 //粒子的生命周期  lifeSpanVariation: 800//生成间隔时间  size: 30//发射的粒子的大小  velocity: PointDirection{ //控制粒子生成的方向  // x:-45 //表示往x轴的负方向生成  x:45 //表示往x轴的正方向生成  // y:-25 //往上生成  //y:25 //往下生成  xVariation: 0  yVariation: 100  }  }  //需要发射的粒子图片  ImageParticle{  id:*particle*  // source: "images/bluecircle.png"  source: "images/hdjlkt.png"  system: *partSys*  colorVariation: 0.3 //随机修改颜色  }  Attractor{  anchors.fill: *parent*  enabled: true  system: *partSys*  pointX: *mar*.mouseX  pointY: *mar*.mouseY  strength: -10000000//该参数为正就是鼠标吸引粒子，为负就是鼠标弹走粒子  //affectedParameter: Attractor.Acceleration  affectedParameter: Attractor.Position //效果较好  // affectedParameter: Attractor.Velocity //也不错  //proportionalToDistance: Attractor.Constant //不好  //proportionalToDistance: Attractor.Linear//不好  // proportionalToDistance: Attractor.Quadratic //不好  // proportionalToDistance: Attractor.InverseLinear  proportionalToDistance: Attractor.InverseQuadratic//效果最好  }  BusyIndicator{  anchors.bottom: *parent*.bottom  anchors.horizontalCenter: *parent*.horizontalCenter  anchors.bottomMargin: 20  } |  |