Create the 8 Queens Applet Shell

1. Create a Java applet named EightQueens

2. implement MouseListener, MouseMotionListener, Runnable, ActionListener (remember to add unimplemented methods)

3. In init() set size to 1020x700

4. Load the queen image  
Use the object declaration: Image m\_imgQueen;  
Then load the image with a media tracker similar to what we used in the floodfill program

5. Now you need to draw the board: add declarations:  
static final int NUMROWS = 8;

static final int NUMCOLS = 8;

static final int SQUAREWIDTH = 50;

static final int SQUAREHEIGHT = 50;

static final int BOARDLEFT = 50;

static final int BOARDTOP = 50;

int m\_nBoard[][] = new int[8][8];

6. Add a paint() method

public void paint (Graphics canvas)

{

m\_bClash = false;

DrawSquares( canvas );

canvas.setColor(Color.RED);

CheckColumns( canvas );

CheckRows( canvas );

CheckDiagonal1( canvas );

CheckDiagonal2( canvas );

canvas.setColor(Color.BLUE);

canvas.drawString(m\_strStatus,

BOARDLEFT, BOARDTOP + SQUAREHEIGHT \* 8 + 20);

}

7. Add the support code:

void DrawSquares( Graphics canvas )

{

canvas.setColor(Color.BLACK);

for( int nRow=0; nRow<NUMROWS; nRow++ )

{

for( int nCol=0; nCol<NUMCOLS; nCol++ )

{

canvas.drawRect( BOARDLEFT + nCol \* SQUAREWIDTH,

BOARDTOP + nRow \* SQUAREHEIGHT, SQUAREWIDTH, SQUAREHEIGHT );

if( m\_nBoard[nRow][nCol] != 0 )

{

canvas.drawImage( m\_imgQueen,

BOARDLEFT + nCol \* SQUAREWIDTH + 8, BOARDTOP + nRow \* SQUAREHEIGHT + 6, null );

}

}

}

}

void CheckColumns( Graphics canvas )

{

// Check all columns

for( int nCol=0; nCol<NUMCOLS; nCol++ )

{

int nColCount = 0;

for( int nRow=0; nRow<NUMROWS; nRow++ )

{

if( m\_nBoard[nRow][nCol] != 0 )

{

nColCount++;

}

}

if( nColCount > 1 )

{

canvas.drawLine( BOARDLEFT + nCol \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + ( SQUAREHEIGHT / 2 ),

BOARDLEFT + nCol \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + SQUAREHEIGHT \* 7 + ( SQUAREHEIGHT / 2 ) );

m\_bClash = true;

}

}

}

void CheckRows( Graphics canvas )

{

for( int nRow=0; nRow<NUMROWS; nRow++ )

{

int nRowCount = 0;

for( int nCol=0; nCol<NUMCOLS; nCol++ )

{

if( m\_nBoard[nRow][nCol] != 0 )

{

nRowCount++;

}

}

if( nRowCount > 1 )

{

canvas.drawLine( BOARDLEFT + ( SQUAREWIDTH / 2 ),

BOARDTOP + nRow \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ),

BOARDLEFT + 7 \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + nRow \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ) );

m\_bClash = true;

}

}

}

void CheckDiagonal1( Graphics canvas )

{

// Check diagonal 1

for( int nRow=NUMROWS-1; nRow>=0; nRow-- )

{

int nCol = 0;

int nThisRow = nRow;

int nThisCol = nCol;

int nColCount = 0;

while( nThisCol < NUMCOLS &&

nThisRow < NUMROWS )

{

if( m\_nBoard[nThisRow][nThisCol] != 0 )

{

nColCount++;

}

nThisCol++;

nThisRow++;

}

if( nColCount > 1 )

{

canvas.drawLine( BOARDLEFT + nCol \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + nRow \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ),

BOARDLEFT + ( nThisCol - 1 ) \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + ( nThisRow - 1 ) \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ) );

m\_bClash = true;

}

}

for( int nCol=1; nCol<NUMCOLS; nCol++)

{

int nRow = 0;

int nThisRow = nRow;

int nThisCol = nCol;

int nColCount = 0;

while( nThisCol < NUMCOLS &&

nThisRow < NUMROWS )

{

if( m\_nBoard[nThisRow][nThisCol] != 0 )

{

nColCount++;

}

nThisCol++;

nThisRow++;

}

if( nColCount > 1 )

{

canvas.drawLine( BOARDLEFT + nCol \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + nRow \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ),

BOARDLEFT + ( nThisCol - 1 ) \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + ( nThisRow - 1 ) \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ) );

m\_bClash = true;

}

}

}

void CheckDiagonal2( Graphics canvas )

{

// Check diagonal 2

for( int nRow=NUMROWS-1; nRow>=0; nRow-- )

{

int nCol = NUMCOLS - 1;

int nThisRow = nRow;

int nThisCol = nCol;

int nColCount = 0;

while( nThisCol >= 0 &&

nThisRow < NUMROWS )

{

if( m\_nBoard[nThisRow][nThisCol] != 0 )

{

nColCount++;

}

nThisCol--;

nThisRow++;

}

if( nColCount > 1 )

{

canvas.drawLine( BOARDLEFT + nCol \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + nRow \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ),

BOARDLEFT + ( nThisCol + 1 ) \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + ( nThisRow - 1 ) \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ) );

m\_bClash = true;

}

}

for( int nCol=NUMCOLS-1; nCol>=0; nCol--)

{

int nRow = 0;

int nThisRow = nRow;

int nThisCol = nCol;

int nColCount = 0;

while( nThisCol >= 0 &&

nThisRow < NUMROWS )

{

if( m\_nBoard[nThisRow][nThisCol] != 0 )

{

nColCount++;

}

nThisCol--;

nThisRow++;

}

if( nColCount > 1 )

{

canvas.drawLine( BOARDLEFT + nCol \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + nRow \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ),

BOARDLEFT + ( nThisCol + 1 ) \* SQUAREWIDTH + ( SQUAREWIDTH / 2 ),

BOARDTOP + ( nThisRow - 1 ) \* SQUAREHEIGHT + ( SQUAREHEIGHT / 2 ) );

m\_bClash = true;

}

}

}