PROJECT: **Create a sleep and wake clock for pre-literate children that runs on mobile platforms (Android and iOS)**

CLIENT: **[self]**

TOOLS: **Eclipse / Flash Builder**
Java / ActionScript 3.0
Flash Professional

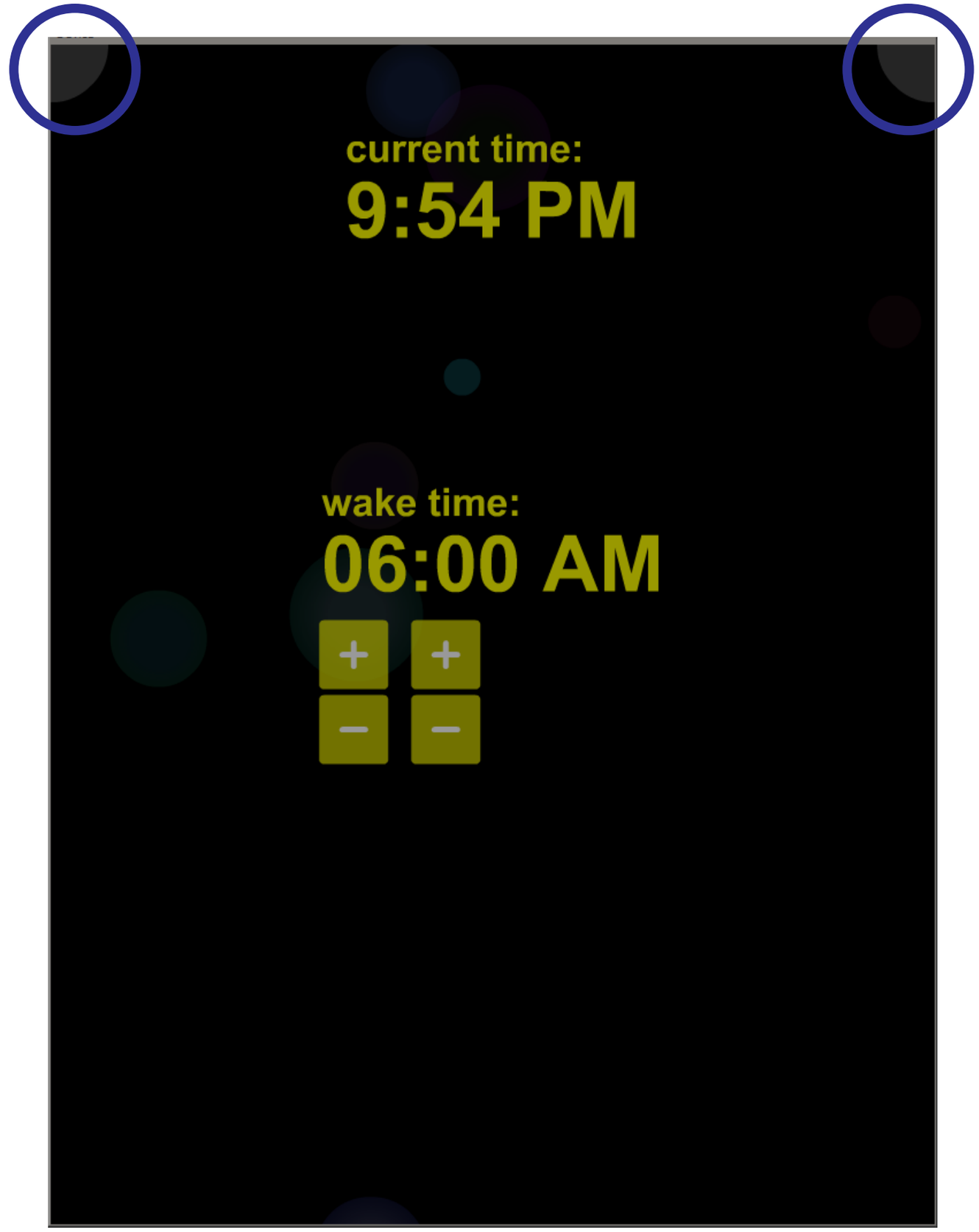
DESCRIPTION: **Very young children often awake before parents. They can not read clocks, so can not determine whether the current time is an acceptable wake time. This clock uses color to indicate time and proximity to wake time, and provides relaxing visuals.**

SLIDES: **5**

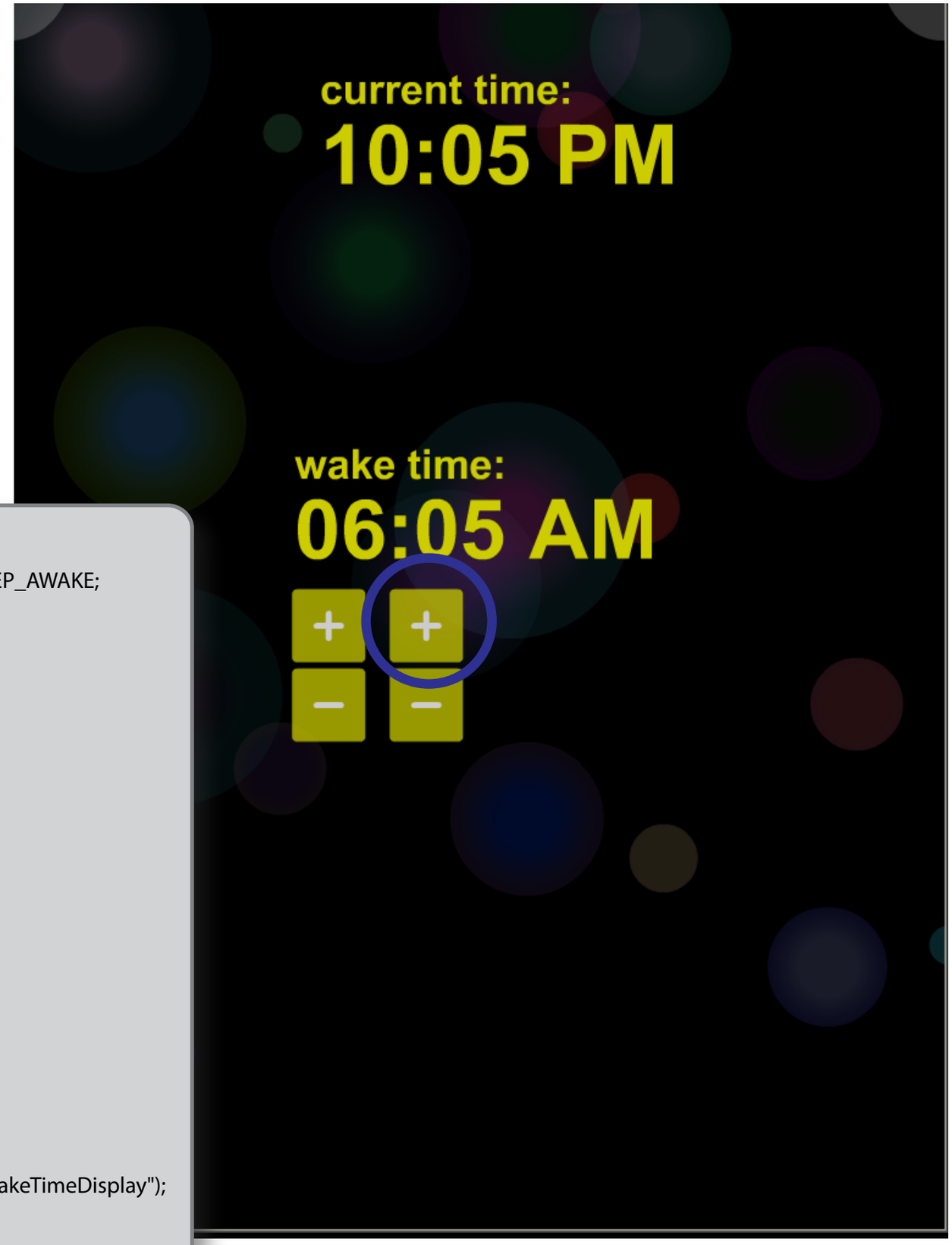
The application launches and renders out the slowly-sliding semi-transparent circles. The ‘bubbles’ drift slowly across the screen, at random speeds, colors, directions and transparency. The bubbles change color as the Wake Time approaches, and when it is reached.

The top-left and top right corners (circled in purple) hold static semi-circles that are actually buttons. They are camoflaged, because toddlers also understand how touchscreens work, and will push any button that is obviously a button.

Top-left spawns and destroys the time/wake interface. Top-right quits the application.



Every screen touch adds one bubble to the background. Bubbles can be added by touching anywhere on the interface. The code structure that permits this involves having the entire interface held in a single object, with a listener assigned to the container object.




```
private function onInit():void
{
        NativeApplication.nativeApplication.systemIdleMode = SystemIdleMode.KEEP_AWAKE;
        Multitouch.inputMode=MultitouchInputMode.TOUCH_POINT;

        addBackground();

        for(var i:int=0; i<numberOfBalls;i++){
                addBouncingBall();
        }

        setChildIndex(bg, numChildren - 1);

        btnCurrentTime = new TimeDisplayButton();
        btnCurrentTime.clock = this;
        btnCurrentTime.addTimeDisplay();
        bg.addChild(btnCurrentTime);

        btnExit = new ExitButton();
        btnExit.x = stage.stageWidth;
        bg.addChild(btnExit);

        bg.addEventListener(MouseEvent.CLICK, addBouncingBall_event);

        wakeTimeDisplay = addWakeTimeDisplayWithControls(wakeTimeDisplay, "wakeTimeDisplay");

        startTimer();

}
```

```
public class BouncingBall extends Sprite
{
	private var ball:Sprite = new Sprite;
	private var startMoving:Boolean = false;
	private var movementX:Number = 1;
	private var movementY:Number = 1;

	private function main(event:Event):void{
		setMovement();
		bounceWalls();
	}

	private function setMovement():void
	{
		if(startMoving == true){
			this.x += movementX;
			this.y += movementY;
		} else {
			this.x -= movementX;
			this.y -= movementY;
		}
	}
```

```
private function bounceWalls():void
{
	if(this.x < 0 || this.y < 0){
		getXAndY();
		startMoving = true;
	}
	if( this.x > clock.bg.width  ||  this.y > 0 + clock.bg.height ){
		getXAndY();
		startMoving = false;
	}
}
```

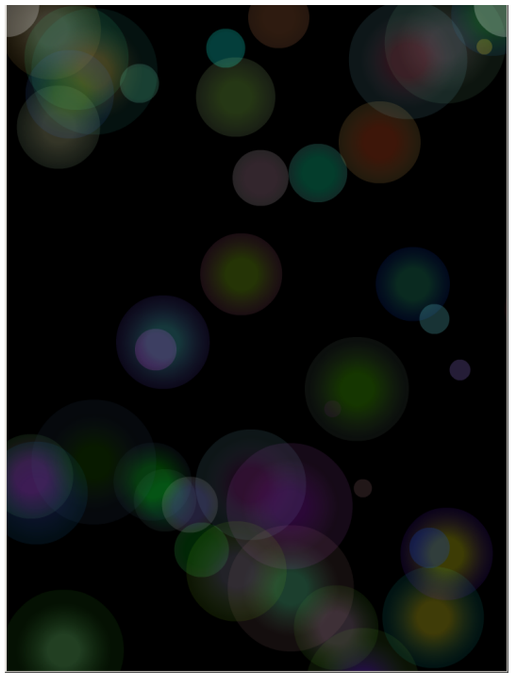
The clock works by changing colors as the wake time approaches. Toddlers may not read numeric clocks, but they understand color as a symbol and can assign meaning: Not Yet / Soon / Wake.

30+ minutes from wake: multicolor

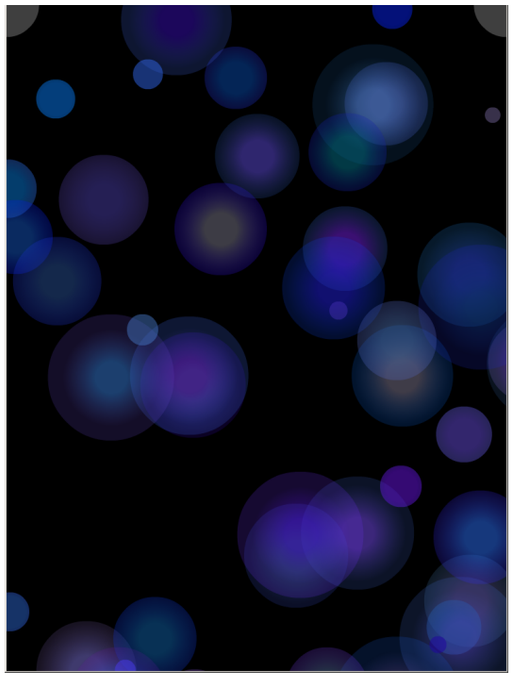
30 to 5 minutes before wake: blue tint

5 to 0 minutes before wake: red tint

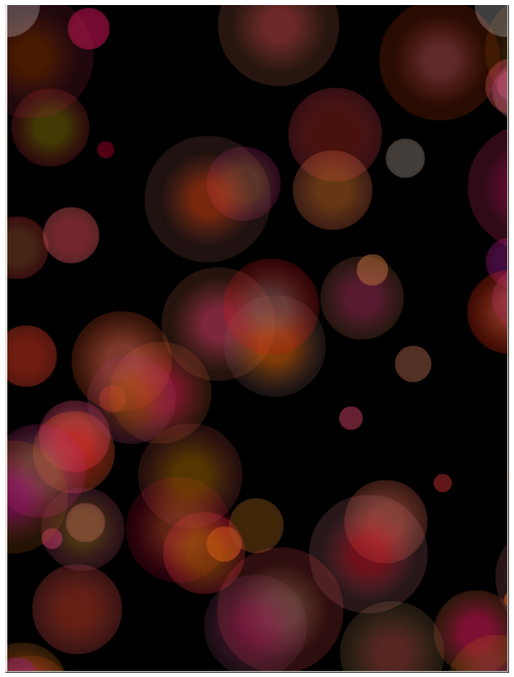
Wake Time: yellow tint



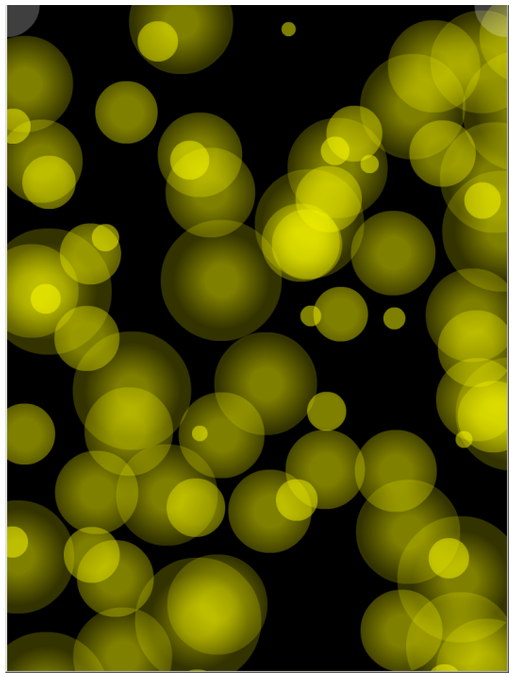
> 30 min



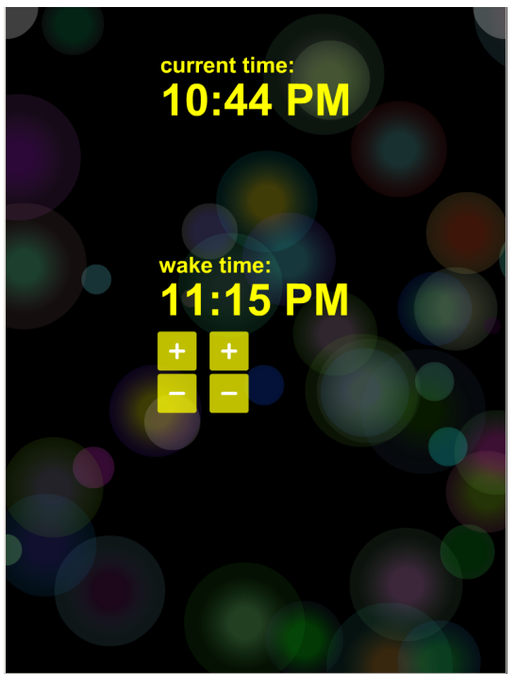
30 to 5 min

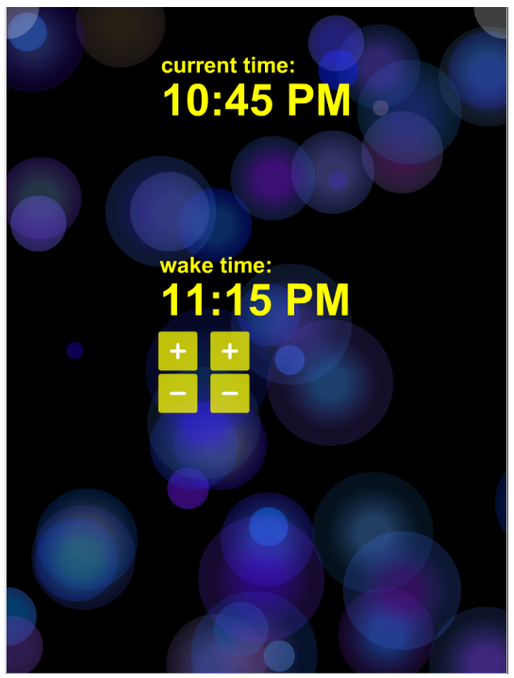


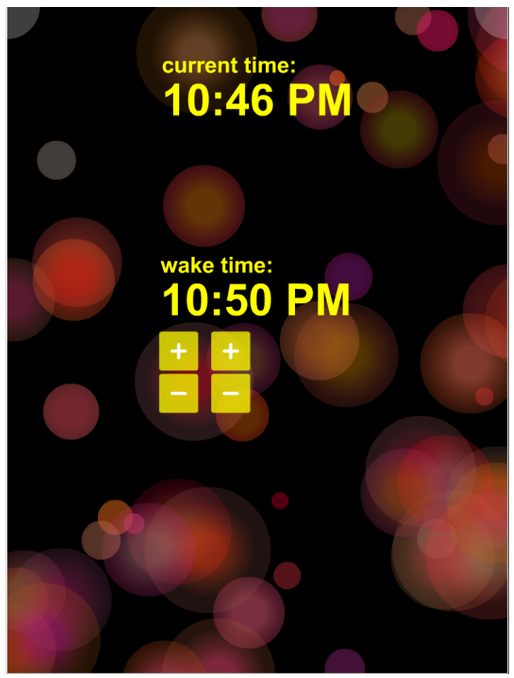
5 to 0 min

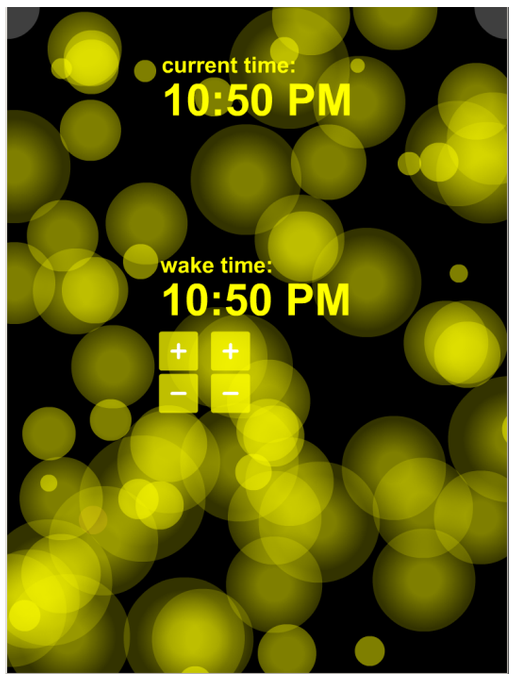


0 min : wake

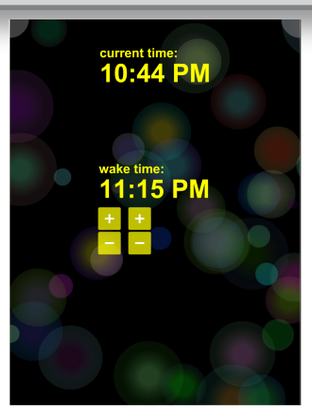








```
public function checkAlarm():void
{
        if(this.alarmOn == true){

                var diff:Number = timeWake.getTime() - timeNow.getTime();
                var i:int;

                if( diff < 30*60*1000 && diff > 5*60*1000){
                        // 30 min * 60 sec * 1000 ms
                        for( i=0;i<numChildren;i++){
                                if(getChildAt(i) is BouncingBall){
                                        tintColor(getChildAt(i) as Sprite, 0x0000FF, 0.5);
                                }
                        }
                }

                if( diff < 5*60*1000 && diff > 0){
                        // 5 min * 60 sec * 1000 ms

                        for( i=0;i<numChildren;i++){
                                if(getChildAt(i) is BouncingBall){
                                        tintColor(getChildAt(i) as Sprite, 0xFF0000, 0.5);
                                }
                        }
                }
                if(timeNow.hours == timeWake.hours && timeNow.minutes == timeWake.minutes){

                        // keep this all in the same date range to manage duration of alarm signal
                        timeWake = timeNow;

                        for( i=0;i<numChildren;i++){
                                if(getChildAt(i) is BouncingBall){
                                        tintColor(getChildAt(i) as Sprite, 0xFFFF00, 1);
                                }
                        }
                }
        }
}
```

else { 

}