

## Main features

The main focus points of the Kalydo game development kit are ease of use and robustness for web games. This results in the following main features;

- Easy to use and powerful engine
- Rapid development
  - GoofBall360 was built within 4 months (800 man hours)
- Top class error handling and reporting
  - Whenever something goes wrong, an error report is sent. This error report includes an error message with line information and all data to replicate the situation on any computer.
- Fully web-optimized
  - Low system specs; 1,2 Ghz pc with a 3D video card and 256MB memory
  - Advanced compression techniques; games are only 5-10MB
  - Server communication for game progress and other user data
- Secure
  - SSL communication, binary and content signing
- Game genre independent
  - Already built a race, shooter, sport and adventure game



## Full list of features

### Overview

- Proven technology (4 games finished, 2 in development)
- Game development fully in Lua scripting
  - Game prototyping within 2 weeks
  - Game development in months instead of years
  - No compiling needed
  - No expensive development environment needed
  - High performance combined with ease of use
- Event based game development
- Web optimized engine
  - Less than 800 KB compressed
  - Low system specs; 1,2 Ghz pc with a 3D video card and 256MB memory
  - Advanced compression techniques
  - Server communication for user data
- Continuous development
  - More features, optimizations and fixes every week

### Security

- SSL communication between web-player and back-end
- Content signing
- Engine and player signing

### Game Objects

- Feature based object creation
  - Build any object with a small set of understandable features

### Rendering

- Focus on compatibility with budget systems
  - 65% of the market uses Intel integrated graphics
  - Runs smoothly on 5 year old pc's
- Fully user controlled lighting system
- Dynamic programmable render sequences



## Animation

- Low cost
- Animation blending in multiple intuitive ways
- Many options
  - Choose your animation clock (forward, backward, zigzag, cosinus, fade, ...)
  - Choose your animation type (model, object, texture, opacity, ...)
  - Choose your settings (how often, how long, speed, ...)
  - Control the animation speed (go back and forth, speed up, slow down)
  - Control the animation (play, stop, pause, ...)
  - Extend the animation (make animation sequences on the fly)

## Physics

- Light weight powerful physics (ODE integration)

## Sound

- OpenAL integration
  - Full 3D sound
  - Sound buffering
  - Sound effects like echo
- Ogg support

## AI system

- Automatic navigation graph generation
  - Allowing dynamic level generation
  - No need to design the navigation graph
- Advanced behavioral system
  - Choose and tweak each behavior separately
  - Many behaviors predefined (follow, seek, avoid, ...)
  - Easily extendible with more behaviors
- Easy to use state machine



## Event system

- Event based
- Advanced job scheduling
  - Jobs can have their own looping behavior
  - Priorities allow for frame-rate stability

## Advanced

- Agent based design
  - Use what you need; unused sub-systems won't take up any time
- Offline building, online testing
- Quaternions for arbitrary rotations
- Advanced error handling system
- Collada support
- Memory pools
- Memory leak detection
- Dangling pointer detection
- Run Time Type Information with member reflection

