Main			
Type	**Name**	**Access**	**Description**
GLFWwindow*	window	main	Pointer for the GLFW window, populated by windowSetup().
render	renderMain	main	Render thread local storage.
shared	sharedMain	main	Shared area object.
render*	renderAP	main	Local storage access pointer.
shared*	sharedAP	main	Shared area access pointer.
std::thread	simThread	main	Object container for simulation thread. Initialised to startup().
GLFWvidmode*	mode	main/windowSetup	Provides access to the current video capability. (Screen Resolution, RGB colour depth, refresh-rate)
int	wXRes	main/windowSetup	Default window X Resolution, screen X*0.8
int	wYRes	main/windowSetup	Default window Y Resolution, screen Y*0.8
control	temp	main/setupDefaultScenario	Tempoary control structure used for setting defaults
std::mutex	simWaitMTX	main/startup	Mutex declared in simulation thread start up, linked to condition_variable simWait to ensure synchronisation.
simulation	simMain	main/startup	Simulation thread local storage.
simulation*	simAP	main/startup	Simulation local store access pointer.
bool	initCalc	main/startup	Boolean, defines if an initial calculation of accelerations / collisions should be carried out.
int	iCount	main/startup	Used in testing for displaying number of iterations since launch.

Body			
Type	**Name**	**Access**	**Description**
double	m	public (body)	Mass of body.
double	r	public (body)	Radius of body.
double	pX	public (body)	X Position of body.
double	pY	public (body)	Y Position of body.
double	vX	public (body)	X Velocity of body.
double	vY	public (body)	Y Velocity of body.
double	aX	public (body)	X Acceleration of body.
double	aY	public (body)	Y Acceleration of body.
bool	fixed	public (body)	Fixed status of body.
float[3]	color	public (body)	RGB colour of body.

Control (Struct)			
Type	**Name**	**Access**	**Description**
double	UGC	public (control)	Gravitational constant.
double	IDT	public (control)	Iteration delta time.
int	IPF	public (control)	Iterations per frame.
bool	collide	public (control)	Collisions switch.
bool	paused	public (control)	Simulation paused.
bool	exit	public (control)	Exit management.

Scenario			
Type	**Name**	**Access**	**Description**
std::vector<body*>	bodies	protected (scenario)	Body pointer storage, bodies are assigned using dynamic allocation.
control	lControl	protected (scenario)	Scenario control data store.

Shared			
Type	**Name**	**Access**	**Description**
std::mutex	bodyLock	private (shared)	Mutex lock for body store.
std::mutex	controlLock	private (shared)	Mutex lock for control store.
std::condition_variable	simWait	public (shared)	Condition variable, locked by sim thread and unlocked by main to synchronise frame display and simulation.

Render			
Type	**Name**	**Access**	**Description**
std::vector<body*>&	pBodies	public (render)	Public facing access for scenario body pointer store for interface access.
control&	pControl	public (render)	Public facing access for scenario control pointer store for interface access.
segments	const int	render::drawBody	Number of segments to be drawn for each body, does not change.
posX	double	render::drawBody	Central position of body X, taken from body pointer passed.
posY	double	render::drawBody	Central position of body Y, taken from body pointer passed.
theta	float	render::drawBody	Constant, 2pi/segments, could be pre-calculated.
tanFact	float	render::drawBody	Constant, tan(theta)
radFact	float	render::drawBody	Constant, cos(theta)
x	float	render::drawBody	Used in circle drawing algorithm for drawing circle, describes the position of the drawn and last drawn points.
y	float	render::drawBody	
lx	float	render::drawBody	
ly	float	render::drawBody	
tempRand	double	render::createSuperstructure	Random number for XY position of bodies.
tempCirX	double	render::createSuperstructure	Body X position in superstructure, constrained by random*cos(2*pi*random)
tempCirY	double	render::createSuperstructure	Body Y position in superstructure, constrained by random*sin(2*pi*random)
tempDist	double	render::createSuperstructure	Holds distance of body from central point.
tempVelX	double	render::createSuperstructure	Holds X velocity for circular orbit for superstructure body.
tempVelY	double	render::createSuperstructure	Holds Y velocity for circular orbit for superstructure body.

Simulation			
Type	**Name**	**Access**	**Description**
double	dX	simulation::calcAcceleration	X component distance between bodies. Force and collision calculations.
double	dY	simulation::calcAcceleration	Y component distance between bodies. Force and collision calculations.
double	dV	simulation::calcAcceleration	Vector distance between bodies. Force and collision calculations.
double	fP	simulation::calcAcceleration	Force pre-component.
double	fX	simulation::calcAcceleration	Force X component.
double	fY	simulation::calcAcceleration	Force Y component.
double	totalMass	simulation::calcAllCollisions	Total mass of bodies in collision.

UI			
Type	**Name**	**Access**	**Description**
render*	g_RenderAP	Global (ui.cpp only)	Global render access pointer for UI, used by GLFW callback functions to access.
TwBar*	simGUI	Global (ui.cpp only)	Pointer for simulation GUI.
TwBar*	bodyGUI	Global (ui.cpp only)	Pointer for body GUI.
TwBar*	ssGUI	Global (ui.cpp only)	Pointer for superstructure GUI.
body*	activeBody	Global (ui.cpp only)	Pointer for the currently selected body, taken from render body store.
int	activeID	Global (ui.cpp only)	Currently selected body id.
int	bodyCount	Global (ui.cpp only)	Current scenario body count.
double	vectX	Global (ui.cpp only)	Vector for mouse drag (X component)
double	vectY	Global (ui.cpp only)	Vector for mouse drag (Y component)
double	scaleFactor	Global (ui.cpp only)	Scale factor, camera zoom.
double	responsiveness	Global (ui.cpp only)	Responsiveness, sensitivity modifier.
int	wX	ui.cpp/setupGUI	Window size X, for ATB.
int	wY	ui.cpp/setupGUI	Window size Y, for ATB.

struct ss		Global (ui.cpp only)	Structure for organisation of superstructure variables for interface.
int	bodies	Public (ss)	Number of outer bodies, +1 for total bodies adding to include central body.
double	cMass	Public (ss)	Mass of central superstructure body.
double	oMass	Public (ss)	Mass of outer superstructure bodies.
double	cRadius	Public (ss)	Radius of central superstructure bodies.
double	oRadius	Public (ss)	Radius of outer superstructure bodies.
double	cPX	Public (ss)	Position of central superstructure body. (X)
double	cPY	Public (ss)	Position of central superstructure body. (Y)
double	cVX	Public (ss)	Velocity of central superstructure body. (X)
double	cVY	Public (ss)	Velocity of central superstructure body. (Y)
double	spacing	Public (ss)	Spacing between central body and outer bodies.
double	radius	Public (ss)	Radius of system.
float[3]	color	Public (ss)	Colour of all bodies in the system.
static bool	checking	ui.cpp/getMouseHeld (static)	Checking is set to true after the first call of get mouse held, preventing the timer from restarting.
static bool	held	ui.cpp/getMouseHeld (static)	Set to true if the timer is now greater than 0.15 seconds.
static double	startTime	ui.cpp/getMouseHeld (static)	The start time of the timer, static declaration means that the data is retained after the function exits and when it is called again.
static double	prevX	ui.cpp/moveCamera (static)	Mouse X in previous call.
static double	prevY	ui.cpp/moveCamera (static)	Mouse Y in previous call.
double	mX	ui.cpp/getCoord	Mouse X
double	mY	ui.cpp/getCoord	Mouse Y
GLint[4]	viewport	ui.cpp/getCoord	Current OpenGL viewport matrix.
GLdouble[16]	modelview	ui.cpp/getCoord	Current OpenGL modelview matrix.
GLdouble[16]	projection	ui.cpp/getCoord	Current OpenGL projection matrix.
GLdouble	ignoreZ	ui.cpp/getCoord	Variable must be passed to be set by gluUnProject for Z axis, not required so ignore.
double	aX	ui.cpp/mouseButtonCallback	Mouse world space X.
double	aY	ui.cpp/mouseButtonCallback	Mouse world space Y.
int	id	ui.cpp/mouseButtonCallback	Selected id taken from checkCoord in renderAP.