UGCC with docker containers

Why?

- OS and user Isolation
- Set memory/CPU limits to the container itself
- Game servers/applications that don't support multitenancy are a perfect use case for containers

Why not?

- Container images, storage and networking considerations add an additional layer of complexity
- Currently must use PID files or Smart PID search for the panel to get the PID of the server
- Container images need to be updated along with host OS patches

Caveats

- Only tested with docker on Linux for the time being
- Container storage is not persistent, so important things need to live outside the container (our templates use -v to mount the server folder within the container)
- EVERYTHING required to run the game server must be present in the container

Special considerations when using Docker with UGCC

- You have to use PID files or Smart PID Search feature within the Panel
    - If using smart PID Search, the executable running within the container needs to have a unique command line argument for the panel to identify it
- Use the --rm flag so the container is removed when it's primary executable is stopped, otherwise you can't start it after stopping
- Must use the -v hostpath:containerpath flag to mount the server within the container
- Use the -p hostport:containerport flag to forward all necessary ports from the host to the container
- Use --name UGCC-%ServerID% flag; not necessary but helps identify which container is which
- Use -w /some/path if you need to set the current working directory within the container

Need an example?

- Our Minecraft docker install config template is a great place to get started. http://redirect.brainless.us/UGCC/autoinstaller/templates/Minecraft-Docker-Linux-installer.ugcc

Install docker

- Great documentation is already available here: https://docs.docker.com/engine/install/

Create docker image

- https://docs.docker.com/engine/reference/builder/
    - Our docker based install templates will do this for you
- You basically need to create a container image that provides everything that is needed

- Could be a monolithic image used for everything, but is sort of against the container mantra
- Could be a server specific image (i.e. a specific image for each game type)