Create 3 Downloads

Download the new Create 3 image:

- http://files.kipr.org/wombat/Wombat-v30.2.0.img

Download firmware for your Create 3:

- https://edu.irobot.com/create3/firmware/H.2.3

You will need a micro SD card reader and a software etching program like:

- Raspberry Pi Imager
- Balena Etcher

Once you have all of these, proceed.

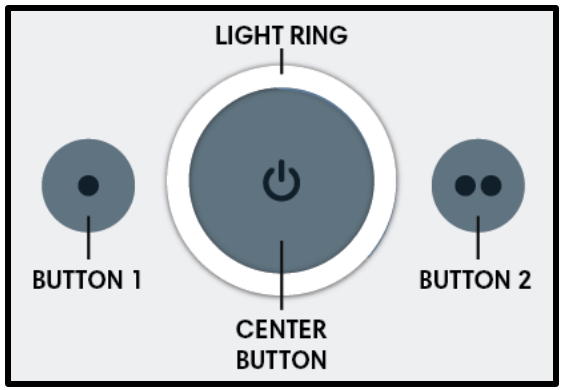
Flashing the SD Card

Backup any programs you want to save to a flash drive or to a computer before proceeding

1. Open the back flap on your Wombat while it is turned off and pull your SD card out.
2. Use your micro SD card reader to connect the SD card to your computer.
3. Using the etching program you downloaded, flash your SD card with the new image. This may take some time as it validates.
4. Once your SD card is flashed, put it back in your Wombat and close the flap.
5. Turn your Wombat on

Setting Up the Create 3

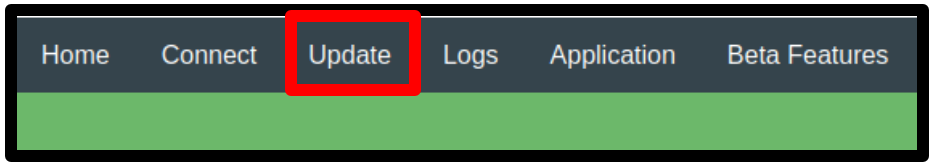
1. Hold down the • and •• buttons on your Create until the light ring shows a spinning blue light.



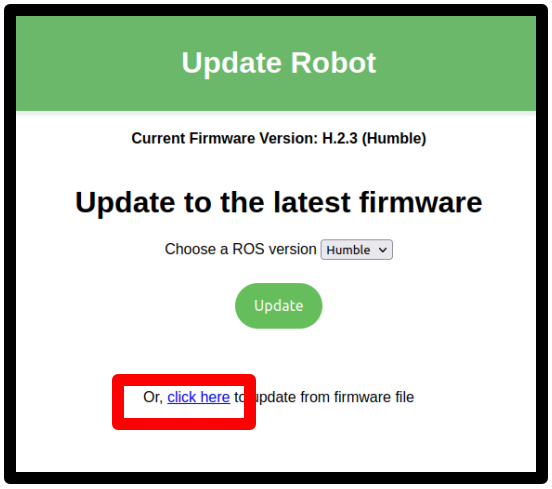
2. Connect to the Create 3 network that shows up with your computer. It will look something like Create-XXX.
3. Once connected, the light ring will stop spinning and show a solid blue.

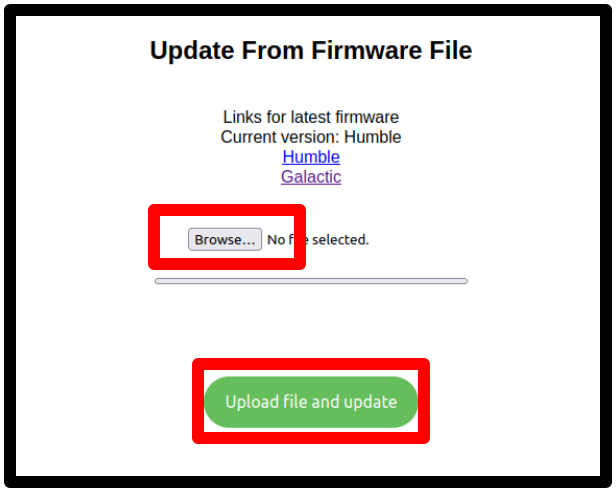
Setting Up the Create 3

1. In a browser on your computer, navigate to: **192.168.10.1**
2. Go to the Update Tab



3. On the update page, select "click here" to update from firmware file. Then browse for your file and click the "Upload file and update" button. It may take some time.

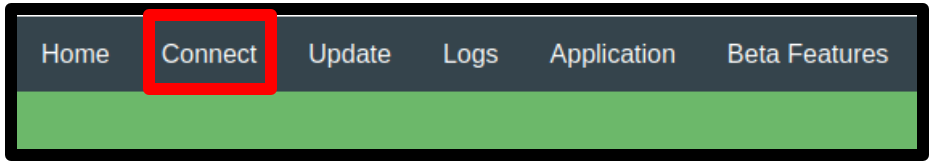




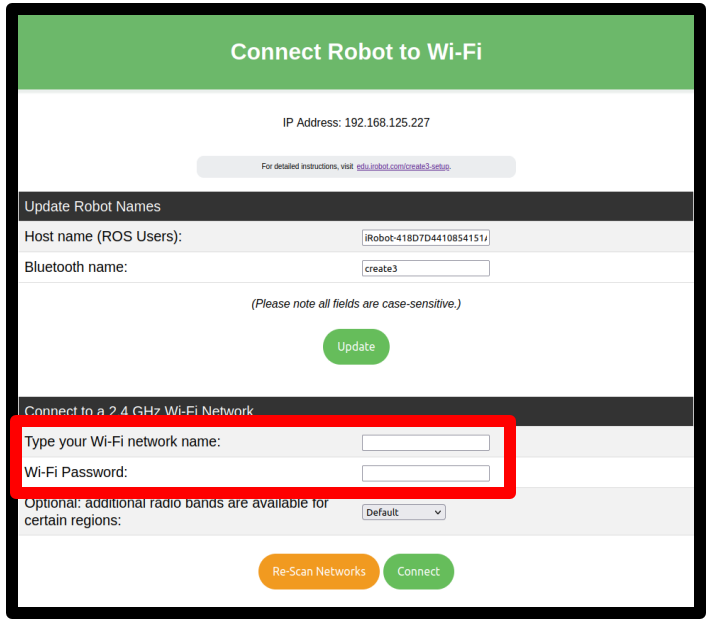
4. The Create will glow solid white and make a happy sound once finished.

Setting Up the Create 3

1. Reconnect to your Create3 and in a browser on your computer, navigate to:
   192.168.10.1
2. Go to the Connect Tab

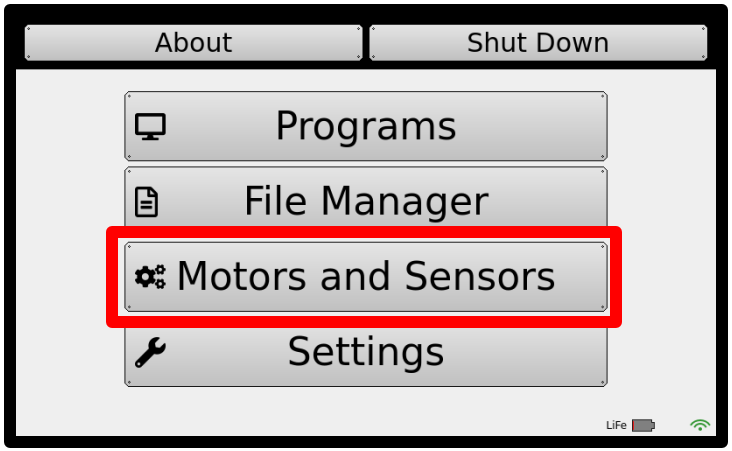


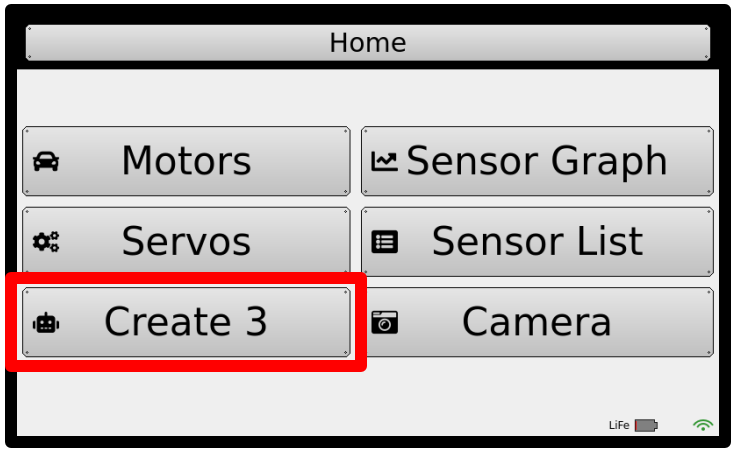
3. Click where the network name is and select your Wombat and then type in the password and click Connect.
4. It may take some time to connect the two.



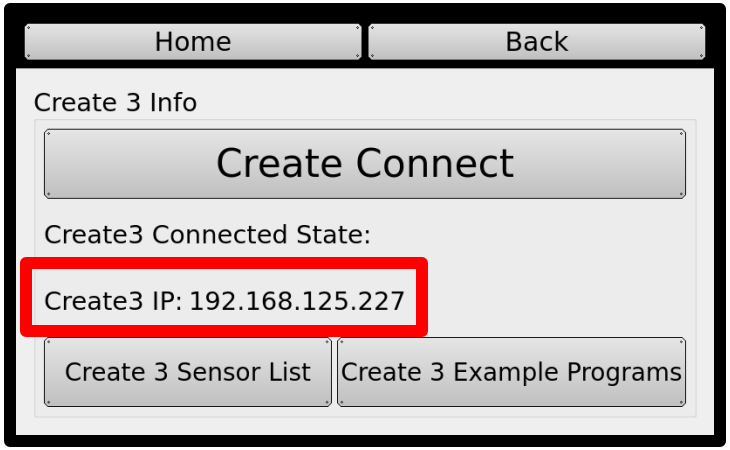
Setting Up the Create 3

1. On your Wombat, go to Motors and Sensors -> Create 3



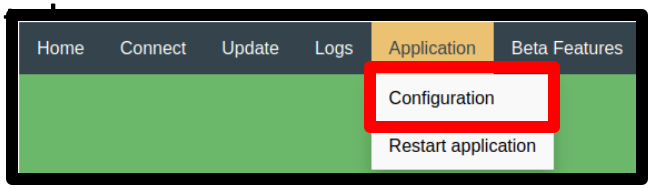


2. On the Create 3 page, look at the Create 3 IP. If you do not see one there, your Create may not be connected to your Wombat yet.

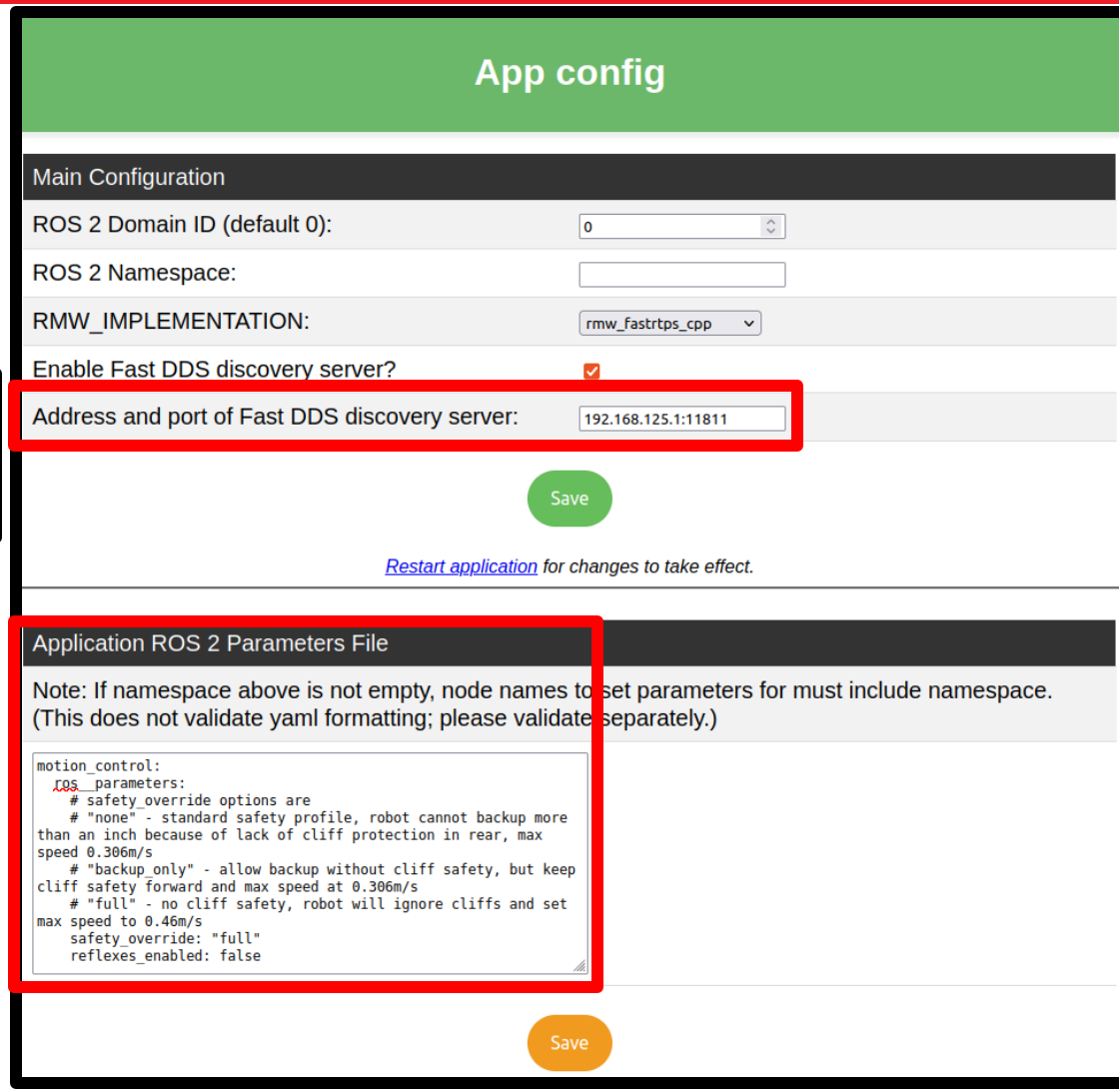


Setting Up the Create 3

1. Go to the IP on your Create 3 page in your browser.
2. Once the Create3 webserver pulls up, go to the Application -> Configuration

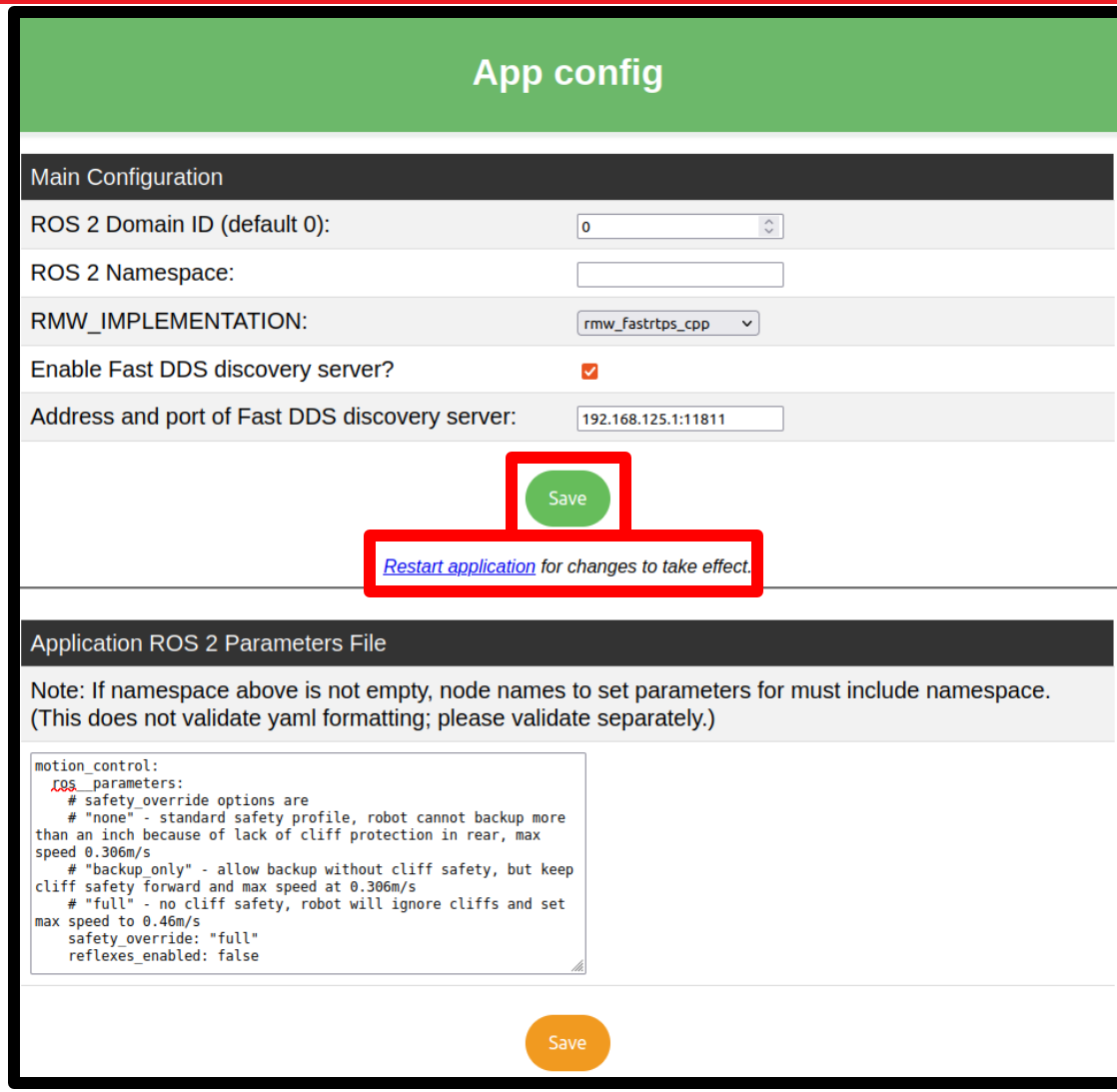


3. Change the "Address and port of Fast DDS discovery server to: **192.168.125.1:11811**
4. Under the ROS 2 Parameters File, change safety_override from **"none"** to **"full"**.
5. Also add: **reflexes_enabled: false**



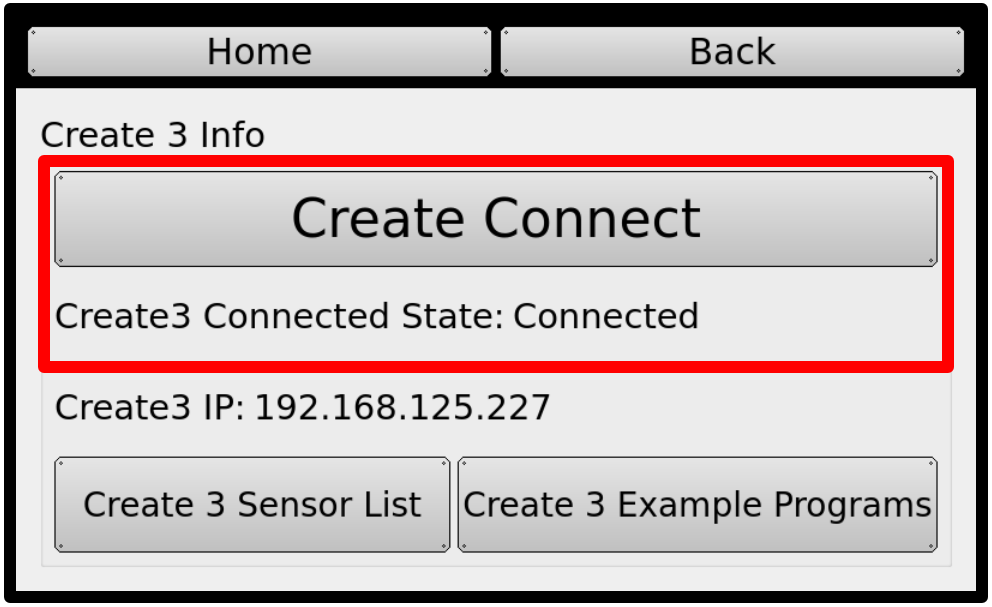
Setting Up the Create 3

1. Once you have made all the changes, Save.
2. Once the page refreshes, comfirm that your changes are there.
3. Click Restart Application.
4. It may take several minutes for the robot to reboot. Once it is finished, it will make a happy sound.



Setting Up the Create 3

1. Reboot your Wombat.
2. Return to the Create 3 page and then Click **Create Connect**.
3. You should see a Connected text next to **Create 3 Connected State**.
4. You can now use Create 3 commands in your code.



Create 3 Function Types

Most Create 3 functions can be split into 3 different types: **Actions**, **Publishers**, and **Subscribers**.

Actions	Publishers	Subscribers
• Sends a goal for the Create 3 to complete (ex. Drive straight, rotate, drive an arc) • Need a **create3_wait()** after a set of commands to execute them • **create3_wait()** tells the program to wait until all queued actions are complete before moving on • Can be pushed past using a **create3_execute_next_command_immediately()** to move on to next command without completing the previous one	• Sends a state for the Create 3 to be in (ex. A specific velocity) • Needs to be repeatedly sent to the Create 3 to keep it in that state	• Receives the state of something from the Create 3 (ex. Cliff sensors, IR sensors, bump sensors, odometry) • Returns a value that can be used by the program

Basic Create 3 Functions

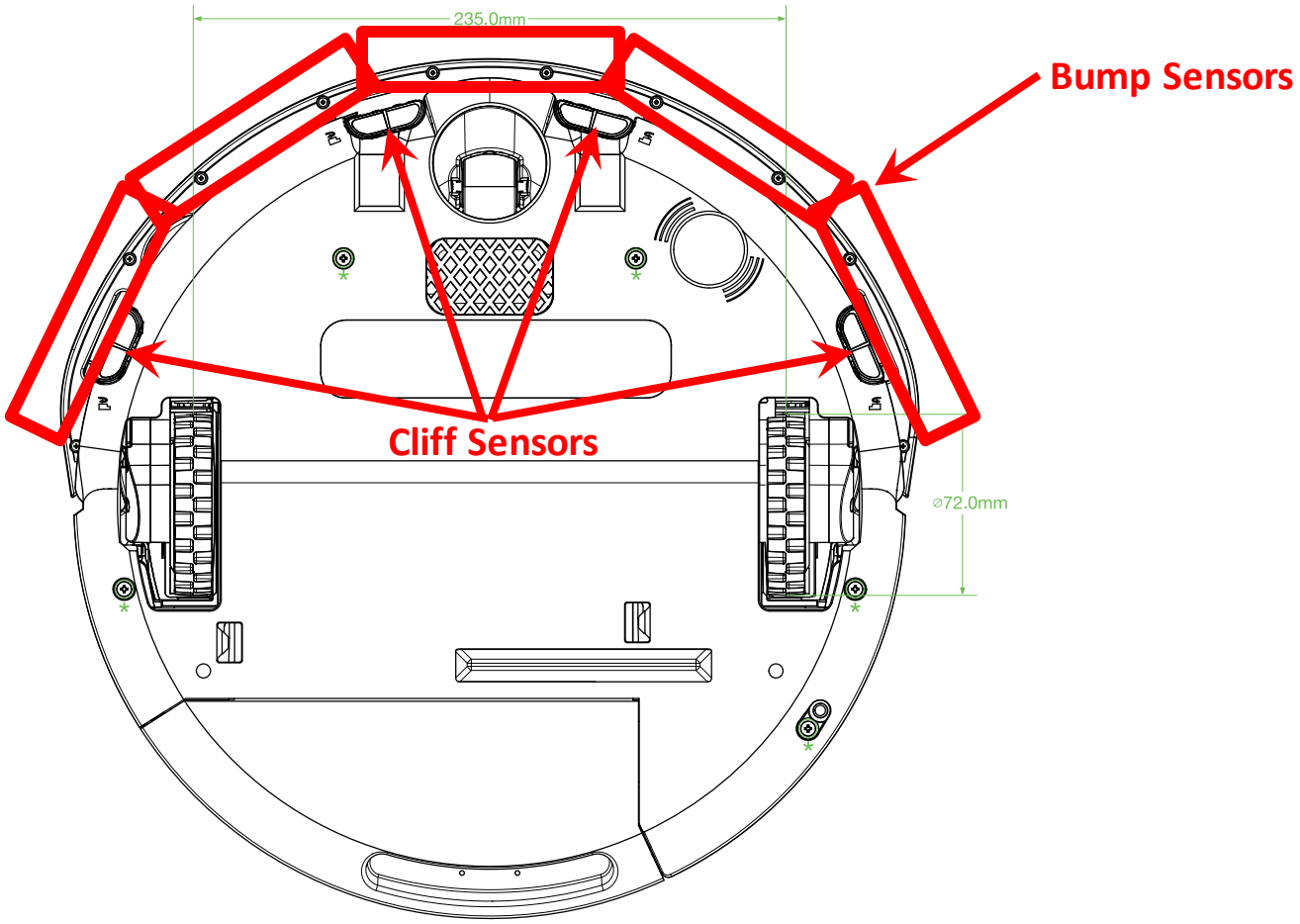
```
int create3_connect();
// Connects to your Create. Returns 1 if successful and 0 if not.

void create3_wait();
// Needed to make any Create 3 actions run. Once called, any actions in the code will run at
this point until they are complete. Other non-Create 3 commands can be run while this is
happening.

void create3_execute_next_command_immediately();
// Runs whatever the next command in the code is immediately even if the previous one has not
completed.

void create3_drive_straight(float distance, float max_linear_speed);
// Action: Drive the Create 3 a distance in meters at a specific speed. The max possible speed
is 0.46 meters/sec. Both arguments can be decimals.

void create3_rotate_degrees(float angle, float max_angular_speed);
void create3_rotate_radians(float angle, float max_angular_speed);
// Action: Rotates the Create 3 a certain number of degrees at a specific speed. Both commands
are the same except the angle units are either degrees or radians and the speeds are degrees/sec
or radians/sec.

void create3_drive_arc_degrees(float radius, float angle, float max_linear_speed);
void create3_drive_arc_radians(float radius, float angle, float max_linear_speed);
// Action: Drives the Create 3 a certain number of degrees along an arc using the radius at a
specific speed. The radius is in meters and the max linear speed is in meters/sec. Both commands
are the same except the angle units are either degrees or radians.
```

Create 3 Sensors

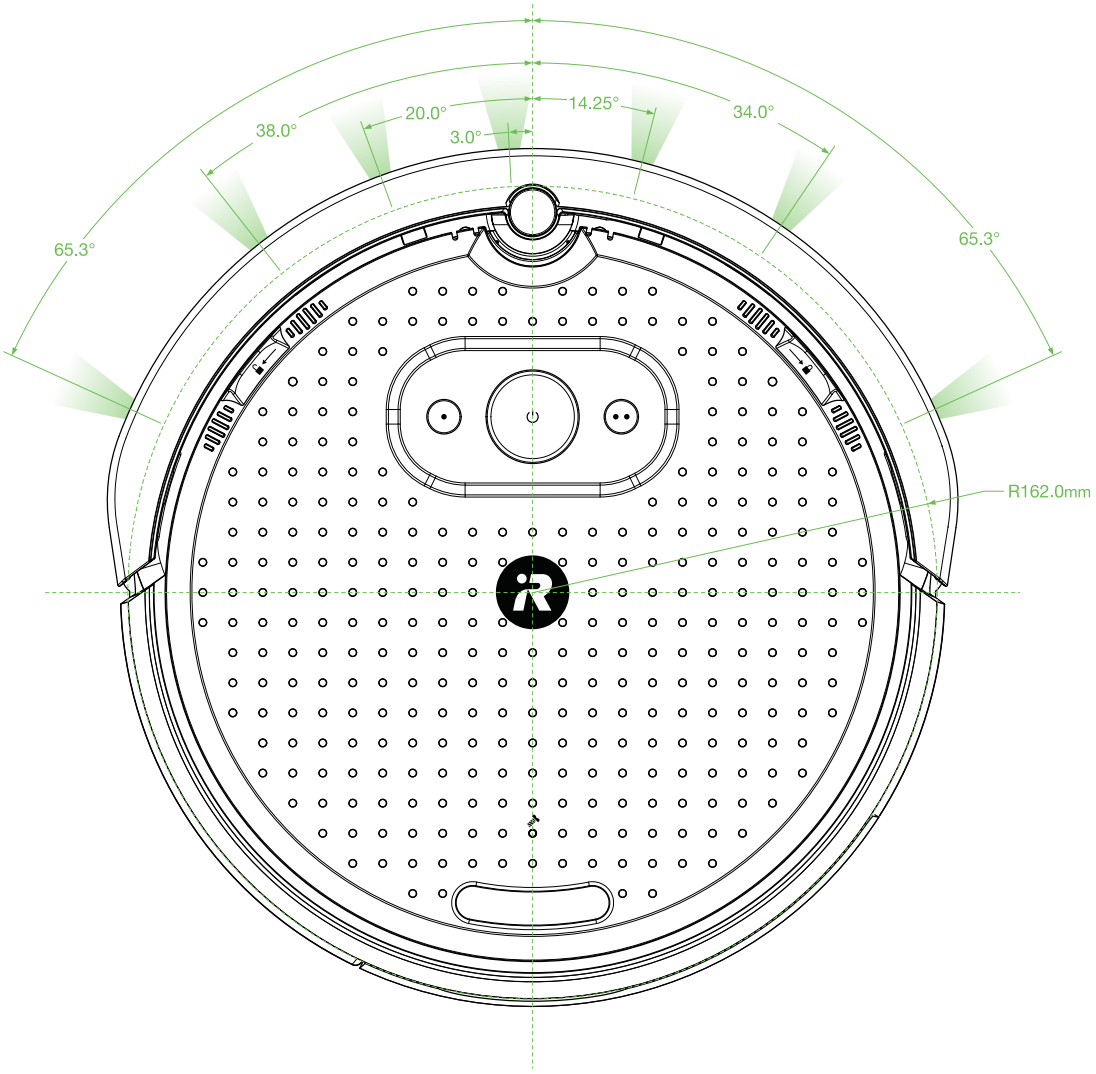
The Create 3 has 4 different types of sensors that can be used. All sensor functions are **subscribers** and will return values from the Create 3. The 4 different sensor types are:

- **Bump sensors** – 5 digital sensors on the front bumper of the Create 3.
- **Cliff sensors** – 4 analog sensors on the bottom of the Create 3. These can detect edges and also the difference between black and white surface.
- **IR Sensors** – 7 analog sensors on the front of the Create 3. These can detect objects close to the Create 3 and give values based on distance.
- **Odometry** – The Create 3 has internal Odometry that can be used to calculate location, orientation, and speeds.

Bump and Cliff Sensor Locations



IR Sensor Locations



Create 3 Sensor Functions

```
int create3_sensor_bump(int sensor_id);
// Returns the value from a bump sensor, either 0 or 1. The sensor ID will be 0-4, with 0 being the furthest left and 4 being the furthest right.

int create3_sensor_cliff(int sensor_id);
// Returns the value from a cliff sensor, with low numbers being for cliffs or edges and higher numbers being for normal surfaces. The sensor ID will be 0-3, with 0 being the furthest left and 3 being the furthest right.

int create3_sensor_ir(int sensor_id);
// Returns the value from an IR sensor, with low numbers being for objects further away and high number being for objects that are closer. The sensor ID will be 0-6, with 0 being the furthest left and 6 being the furthest right.

double create3_get_euler_z();
// Returns the orientation about the Z axis in radians. The same function can be called but for x or y. However, the Create most likely won't be in a situation where it is rotated about either of those axes.

Create3Pose create3_pose_get();
// Returns a Create3Pose which is a struct containing a Create3Vector3 that holds the current position and a quaternion which holds the current orientation.
```

Create 3 Velocity Functions

```
void create3_velocity_set_components(double linear_x, double angular_z);
// Publisher: Sets the Create 3 velocity in the linear x and it's rotational speed in the angular z.

double create3_velocity_get_angular_z();
// Returns the current velocity about the z axis. This is useful for detecting speed during rotations or arcs.

double create3_velocity_get_linear_x();
// Returns the current velocity in the linear x direction. This is useful for when the Create is driving straight forward.

Create3Twist create3_velocity_get();
// Returns a Create3 Twist which is a struct containing a linear x velocity and an angular z velocity.

Create3Odometry create3_odometry_get();
// Returns a Create3Odometry which is a struct containing a Create3Pose and a Create3Twist. A Create3Pose contains a Create3Vector3 position and a Create3 quaternion. A Create3Twist contains a linear x velocity and an angular z velocity.
```

Create 3 Light Ring

```
Create3LedColor create3_led_color(int r, int g, int b);
// Creates a Create3 LED color struct which can be used in other light ring functions using typical r, g, b values.

Create3LightRing create3_led_color(
    Create3LedColor led0,
    Create3LedColor led1,
    Create3LedColor led2,
    Create3LedColor led3,
    Create3LedColor led4,
    Create3LedColor led5,
);
// Creates a Create3 Light Ring struct which can be used in other light ring functions. Each led can be set to a color. If you are using one color you can set them all to the same variable made using create3_led_color.

void create3_led_animation(
    Create3LedAnimationType animation_type,
    Create3LightRing lightring,
    double max_runtime
);
// Sets a Create3 LED animation to run for a specified amount of time. The two animation types are Create3BlinkLights and Create3SpinLights and must be typed like that for the animation_type argument. The Create3LightRing can be made using other light ring functions. The time is in seconds and can be specified down to the decimal. This command is good for learning structs as well as creating indicators during your code.
```

Create 3 Velocity Functions

```
void create3_velocity_set_components(double linear_x, double angular_z);
// Publisher: Sets the Create 3 velocity in the linear x and it's rotational speed in the angular z.

double create3_velocity_get_angular_z();
// Returns the current velocity about the z axis. This is useful for detecting speed during rotations or arcs.

double create3_velocity_get_linear_x();
// Returns the current velocity in the linear x direction. This is useful for when the Create is driving straight forward.

Create3Twist create3_velocity_get();
// Returns a Create3 Twist which is a struct containing a linear x velocity and an angular z velocity.

Create3Odometry create3_odometry_get();
// Returns a Create3Odometry which is a struct containing a Create3Pose and a Create3Twist. A Create3Pose contains a Create3Vector3 position and a Create3 quaternion. A Create3Twist contains a linear x velocity and an angular z velocity.
```

Building on the Create

The Create 3 has two different surfaces where things may be attached. All hole spacing is 12mm so LEGO holes will line up every other hole.

- Faceplate – The top surface of the Create 3. It can be twisted counter-clockwise to remove it and attach a payload.
- Internal Cargo Bay – Can be removed from the back to attach a payload inside.

All holes on the Create can have items attached using M3 screws (those are the same ones used to attach lever sensors to robots).

These holes may be drilled out to use larger screws at the user's own expense. Replacement faceplates may be purchased from iRobot.