|  |  |
| --- | --- |
| Project | **Standard for Actuator Interface for Cyber and Physical World**<https://sagroups.ieee.org/2888/ **>** |
| Title | **Application Programming Interfaces for Environmental Change Related Actuators** |
| DCN | **2888-22-0022-00-0002** |
| Date Submitted | **Feb. 14, 2022** |
| Source(s) | Yegi Lee zxcasd312@naver.com (Konkuk University)Shin Kim new.xin22@gmail.com (Konkuk University)Kyoungro Yoon yoonk@konkuk.ac.kr (Konkuk University) |
| Re: |  |
| Abstract | This contribution proposes the application programming interfaces for environmental change related actuators |
| Purpose | To start discussion on purpose of the standard |
| Notice | This document has been prepared to assist the IEEE 2888 Working Group. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein. |
| Release | The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE’s name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE’s sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that IEEE 2888 may make this contribution public. |
| Patent Policy | The contributor is familiar with IEEE patent policy, as stated in [Section 6 of the IEEE-SA Standards Board bylaws](http://standards.ieee.org/guides/opman/sect6.html#6.3) <[http://standards.ieee.org/guides/bylaws/sect6-7.html#6](http://127.0.0.1:4664/cache?event_id=757737&schema_id=1&s=5X0vID10lu_E6yrIkWkNd4Wz2H8&q=hancock)> and in *Understanding Patent Issues During IEEE Standards Development* <http://standards.ieee.org/board/pat/faq.pdf> |

# Introduction

This contribution proposes the application programming interfaces for environmental change related actuators.

Environmental change related actuators include:

* Sprayer actuator
* Fog actuator
* Wind actuator
* Bubble actuator

# API for individual actuators

Table 1 – Sprayer API

|  |
| --- |
| Nested Classes |
| Modifier and Type | Method and Description |
|  |  |
| Constructor |
| Constructor and Description |
| Sprayer() |
| *Default constructor.* |
|  |
| Sprayer(String id) |
|  |
| Sprayer(String id, String serverIPAddress, integer serverPort) |
|  |
| Fields |
| Modifier and Type | Field and Description |
|  |  |
| Methods |
| Modifier and Type | Method and Description |
| int | setSprayerComplexCommand(sprayerType sprayer, int intensity) |
|  | *This function sets a command to designate a sprayed material type and the intensity of sprayer actuator. This function shall include a sprayer parameter defined by sprayerType from IEEE2888.2(Section 4.2.5) with intensity parameter. If the command succeeds, this function returns 1; otherwise, it returns 0.* |
|  |  |
| int | setSprayerType(sprayerType sprayer) |
|  | *This function sets a command to designate a sprayed material type of sprayer actuator. This function shall include a sprayer parameter defined by sprayerType from IEEE2888.2(Section 4.2.5). If the command succeeds, this function returns 1; otherwise, it returns 0.* |
|  |  |
| Int | setSprayerIntensity(int intensity) |
|  | *This function sets a command for control intensity of scent emission on scent actuator. This function shall include the intensity parameter. If the command succeeds, this function returns 1; otherwise, it returns 0.* |
|  |  |

Table 2 – Fog API

|  |
| --- |
| Nested Classes |
| Modifier and Type | Method and Description |
|  |  |
| Constructor |
| Constructor and Description |
| Fog() |
| *Default constructor.* |
|  |
| Fog(String id) |
|  |
| Fog(String id, String serverIPAddress, integer serverPort) |
|  |
| Fields |
| Modifier and Type | Field and Description |
|  |  |
| Methods |
| Modifier and Type | Method and Description |
| int | setFogIntensity (int intensity) |
|  | *This function sets a command to control the intensity of fog emissions on the fog actuator. This function shall include the intensity parameter. If the command succeeds, this function returns 1; otherwise, it returns 0.* |
|  |  |

Table 3 – Wind API

|  |
| --- |
| Nested Classes |
| Modifier and Type | Method and Description |
|  |  |
| Constructor |
| Constructor and Description |
| Wind() |
| *Default constructor.* |
|  |
| Wind(String id) |
|  |
| Wind(String id, String serverIPAddress, integer serverPort) |
|  |
| Fields |
| Modifier and Type | Field and Description |
|  |  |
| Methods |
| Modifier and Type | Method and Description |
| int | setWindIntensity (int intensity) |
|  | *This function sets a command to control the intensity of wind on the wind actuator. This function shall include the intensity parameter. If the command succeeds, this function returns 1; otherwise, it returns 0.* |
|  |  |

Table 4 – Bubble API

|  |
| --- |
| Nested Classes |
| Modifier and Type | Method and Description |
|  |  |
| Constructor |
| Constructor and Description |
| Bubble() |
| *Default constructor.* |
|  |
| Bubble(String id) |
|  |
| Bubble(String id, String serverIPAddress, integer serverPort) |
|  |
| Fields |
| Modifier and Type | Field and Description |
|  |  |
| Methods |
| Modifier and Type | Method and Description |
| int | setBubble() |
|  | *This function sets a command to turn on the wind actuator. If the command succeeds, this function returns 1; otherwise, it returns 0.* |
|  |  |