TIZEN™

Tizen.IoTivity.Connectivity Client

GeunSun Lee

Tizen Talks

TIZEN

position-finder-client

Git path : apps/native/position-finder-client
Branch : master, dotnet

index : apps/native/position-finder-client

Domain: Applications; Licenses: Apache-2.0

summary refs log tree commit diff

Branch	Commit message
cs	Add ignore file for visual studio project
dotnet	1. Expand queue size: 20 -> 31
master	Merge "add functions for using rest api"

Age	Commit message
5 hours	Merge "add functions for using rest api" HEAD master
5 hours	add functions for using rest api refs/changes/01/145001/2
5 hours	Add routines for error check refs/changes/96/144996/1
3 days	Add values, then send the sum refs/changes/31/144931/1
4 days	Add resource & controller features refs/changes/10/144710/1
4 days	remove a wrong path in the spec refs/changes/93/144693/1
4 days	Move CBOR file from apps_rw to app's res dir refs/changes/81/144681/1
4 days	Merge with bangbang-sensor-pi refs/changes/79/144679/1
4 days	Merge with bangbang-collector-pi refs/changes/99/144199/5
2017-08-03	Retry to connect servers every 5 sec refs/changes/62/142262/3
[...]	

Clone
https://git.tizen.org/cgit/apps/native/position-finder-client
git://git.tizen.org/apps/native/position-finder-client

TODO

1. Server 찾기
2. Server에서 값 가져오기
3. Remote Resource 컨트롤

"org.tizen.door"

```
int connectivity_observe_resource(
        const char *type,
        connectivity_observe_resource_cb cb,
        void *user_data);
```

```
typedef void (*connectivity_observe_resource_cb)(connectivity_resource_s *resource_info, void *resource_data, void *user_data);
```

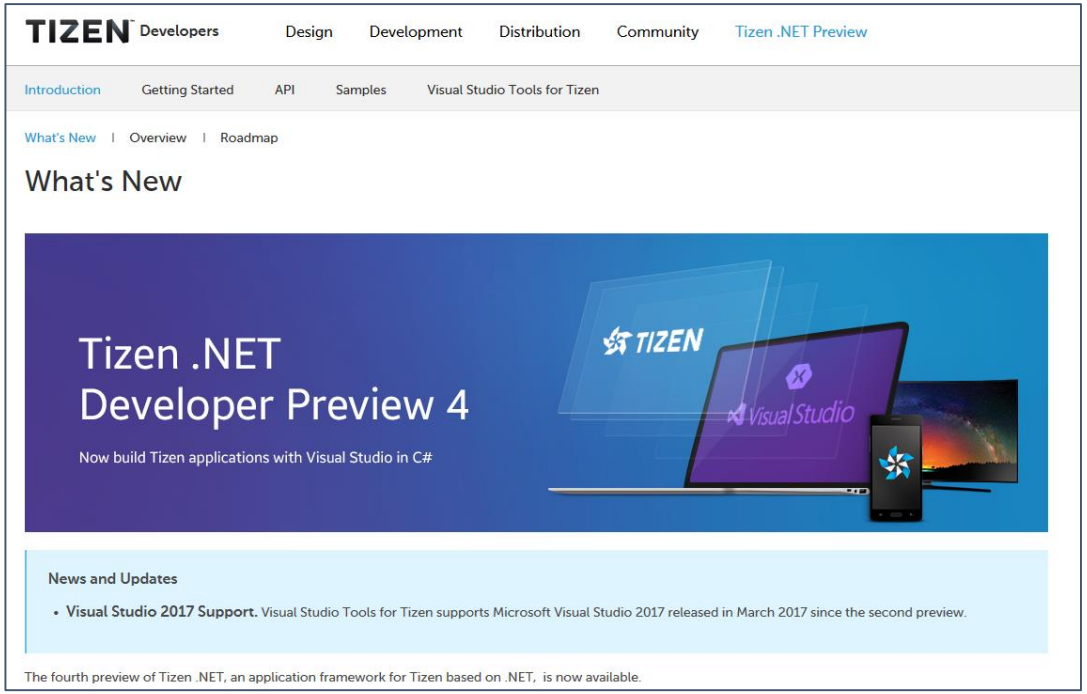
Server & Client

TIZEN .NET

https://goo.gl/mHa2Dw

TIZEN

Tizen .NET



Tizen Talks

TIZEN

https://goo.gl/asf9n3

TIZEN

Tizen .NET

Finding Resources

To find resources:

1. To find a resource, call the `StartFindingResource()` method of the Tizen.Network.IoTConnectivity.IoTConnectivityClientManager class:

```
int RequestId = -1;

ResourceQuery query = new ResourceQuery();
query.Type = "oic.iot.door";

RequestId = IoTConnectivityClientManager.StartFindingResource
(IoTConnectivityClientManager.MulticastAddress, query);
```

2. To get the remote resource handle information, use an event handler registered for the `ResourceFound` event of the `Tizen.Network.IoTConnectivity.IoTConnectivityClientManager` class before calling the `StartFindingResource()` method:

```
ResourceFoundEventArgs outArgs = null;

EventHandler<ResourceFoundEventArgs> handler = null;
EventHandler<FindingErrorOccurredEventArgs> errorHandler = null;

handler = (sender, e) =>
{
    Log.Info(LOGTAG, "ResourceFound:" + e.RequestId + ", HostAddress:" +
e.Resource.HostAddress + ", UriPath:" + e.Resource.UriPath);
    IoTConnectivityClientManager.ResourceFound -= handler;
    IoTConnectivityClientManager.FindingErrorOccurred -= errorHandler;
    outArgs = e;
};
```

Tizen Talks

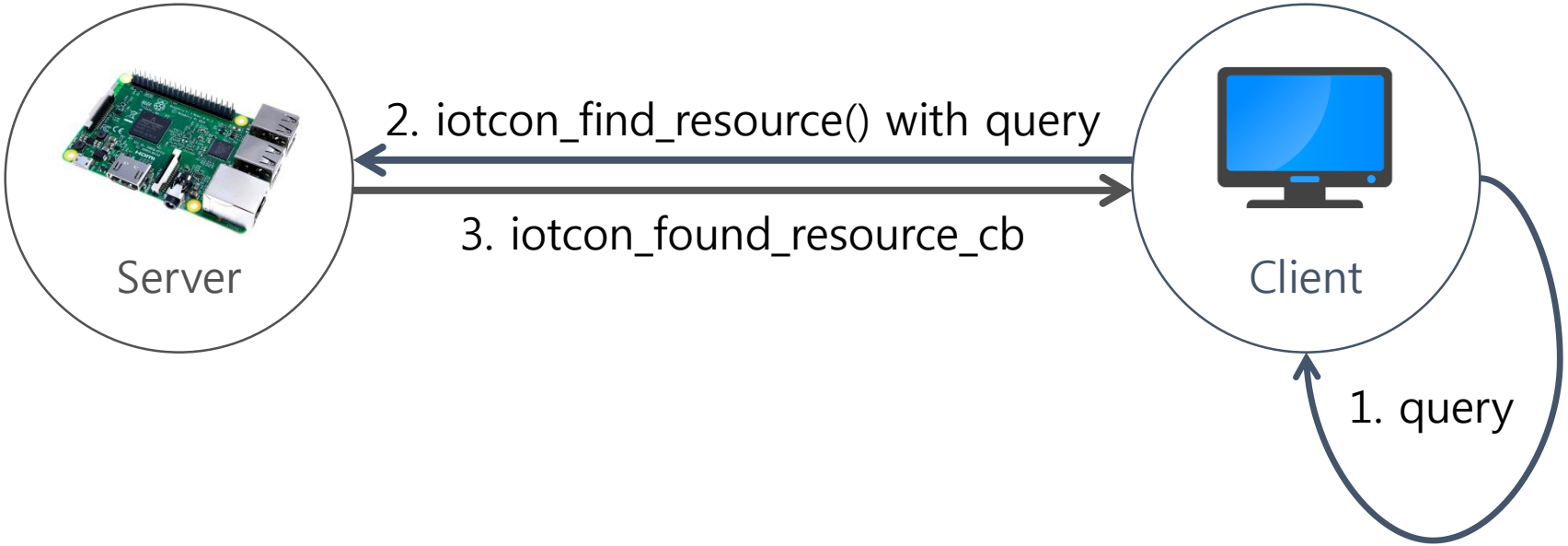
TIZEN

APPENDIX

Client

FIND SERVER

Find Server


Server
2. iotcon_find_resource() with query
3. iotcon_found_resource_cb
Client
1. query

QUERY

```
int  iotcon_query_create (iotcon_query_h *query)
int  iotcon_query_destroy (iotcon_query_h query)

int  iotcon_query_get_resource_type (iotcon_query_h query, char **resource_type)
int  iotcon_query_get_interface (iotcon_query_h query, char **resource_iface)
int  iotcon_query_set_resource_type (iotcon_query_h query, const char *resource_type)
int  iotcon_query_set_interface (iotcon_query_h query, const char *resource_iface)

int  iotcon_query_add (iotcon_query_h query, const char *key, const char *value)
int  iotcon_query_remove (iotcon_query_h query, const char *key)
int  iotcon_query_lookup (iotcon_query_h query, const char *key, char **data)

int  iotcon_query_foreach (iotcon_query_h query, iotcon_query_foreach_cb cb, void *user_data)
```

https://goo.gl/4dYfbY

Find Resource

```
int  iotcon_find_resource (const char *host_address,
            int connectivity_type,
            iotcon_query_h query,
            iotcon_found_resource_cb cb,
            void *user_data)

typedef bool(*  iotcon_found_resource_cb )(iotcon_remote_resource_h resource,
            iotcon_error_e result,
            void *user_data)
```

server

```
int connectivity_set_resource(const char *uri_path, const char *type,
connectivity_resource_s **out_resource_info)
{
    ...
    iotcon_resource_interfaces_h ifaces = NULL;
    ...

    ret = iotcon_resource_interfaces_add(ifaces,
                    IOTCON_INTERFACE_DEFAULT);
    goto_if(IOTCON_ERROR_NONE != ret, error);

    ret = iotcon_resource_interfaces_add(ifaces,
                    IOTCON_INTERFACE_BATCH);
    goto_if(IOTCON_ERROR_NONE != ret, error);


    ...

    ret = iotcon_resource_create(uri_path,
            resource_types,
            ifaces,
            policies,
            _request_resource_handler,
            resource_info,
            &resource_info->res);
    goto_if(IOTCON_ERROR_NONE != ret, error);
    ...
}
```

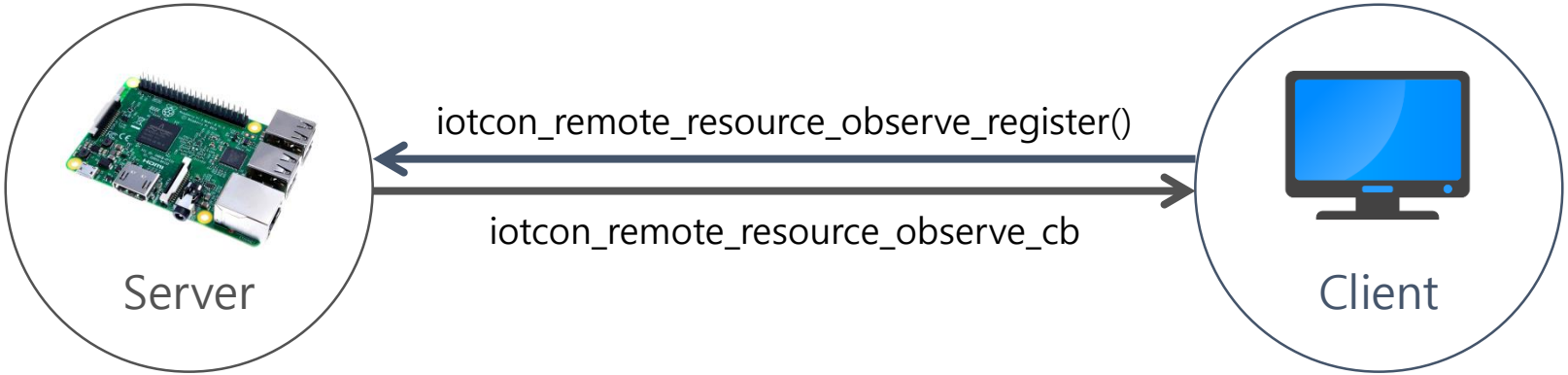
client

```
int connectivity_observe_resource(connectivity_observe_resource_cb cb,
void *user_data)
{
    ...
     ret = iotcon_find_resource(IOTCON_MULTICAST_ADDRESS,
            IOTCON_CONNECTIVITY_IP |
                    IOTCON_CONNECTIVITY_PREFER_UDP,
            query,
            _found_resource_cb,
            cb_info);
    goto_if(IOTCON_ERROR_NONE != ret, error);
    ...
}

static bool _found_resource_cb(iotcon_remote_resource_h resource,
iotcon_error_e result, void *user_data)
{
    ...
    iotcon_resource_interfaces_h resource_interfaces;
    ...
    ret = iotcon_remote_resource_get_interfaces(resource,
&resource_interfaces);
    retv_if(IOTCON_ERROR_NONE != ret, -1);

    ret = iotcon_resource_interfaces_foreach(resource_interfaces,
_get_res_iface_cb, uri_path);
    retv_if(IOTCON_ERROR_NONE != ret, -1);
}
```

Server & Client

OBSERVE

Observe remote resources

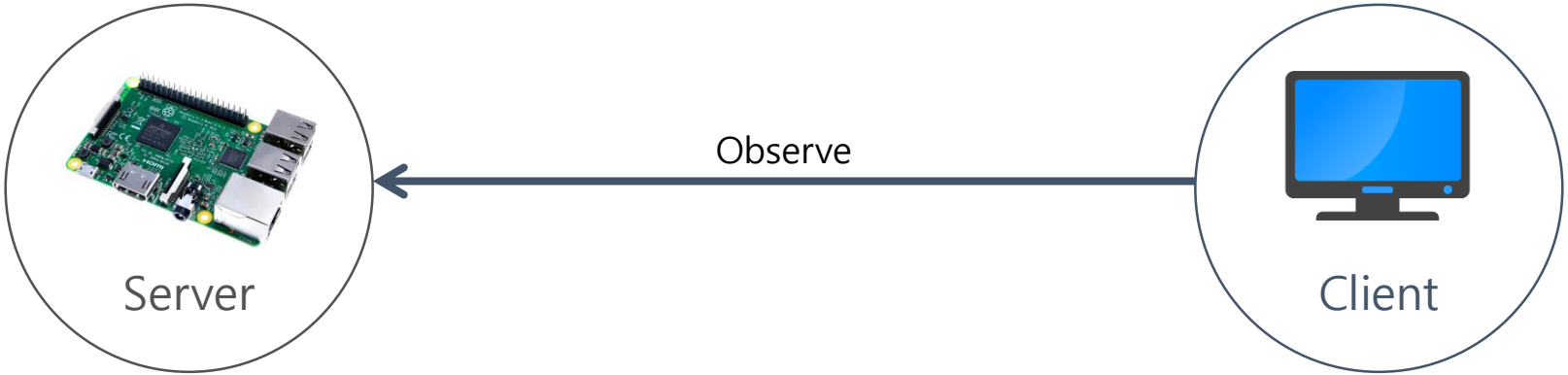


Observe remote resources

```
int  iotcon_remote_resource_observe_register (iotcon_remote_resource_h resource,
          iotcon_observe_policy_e observe_policy,
          iotcon_query_h query,
          iotcon_remote_resource_observe_cb cb,
          void *user_data)

int  iotcon_remote_resource_observe_deregister (iotcon_remote_resource_h resource)
```

IOTCON_OBSERVE_IGNORE_OUT_OF_ORDER
IOTCON_OBSERVE_ACCEPT_OUT_OF_ORDER

Observers




```
int  iotcon_observers_create (iotcon_observers_h *observers)
int  iotcon_observers_destroy (iotcon_observers_h observers)

int  iotcon_observers_add (iotcon_observers_h observers, int obs_id)
int  iotcon_observers_remove (iotcon_observers_h observers, int obs_id)
```

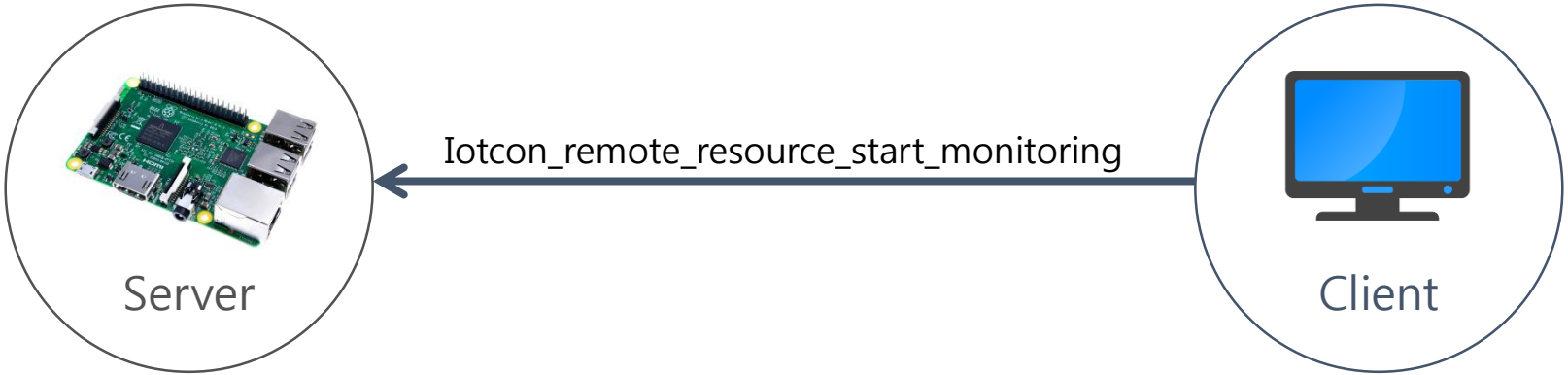
server

```
int connectivity_set_resource(const char *uri_path, const char *type,
connectivity_resource_s **out_resource_info)
{
    ...
    iotcon_resource_create(uri_path,
            resource_types, ifaces, policies, _request_resource_handler,
            resource_info, &resource_info->res);
    ...
}

static void _request_resource_handler(iotcon_resource_h resource, iotcon_request_h
request, void *user_data)
{
    iotcon_request_get_observe_type(request, &observe_type);
    if (IOTCON_OBSERVE_REGISTER == observe_type) {
        iotcon_request_get_observe_id(request, &observe_id);
        iotcon_observers_add(observers, observe_id);
    } else if (IOTCON_OBSERVE_DEREGISTER == observe_type) {
        iotcon_request_get_observe_id(request, &observe_id);
        iotcon_observers_remove(observers, observe_id);
    }

    return 0;
}
```

client

```
int connectivity_observe_resource(... , connectivity_observe_re
{
    ...
    ret = iotcon_find_resource(IOTCON_MULTICAST_ADDRESS,
            IOTCON_CONNECTIVITY_IP | IOTCON_CONNECTIVITY_PREFER_UDP,
            query,
            _found_resource_cb,
            cb_info);
    ...
}
static bool _found_resource_cb(...)
{
    ...
    iotcon_remote_resource_observe_register(info->resource,
            IOTCON_OBSERVE_IGNORE_OUT_OF_ORDER,
            NULL,
            _observe_cb,
            info);
    ...
}
static void _observe_cb(iotcon_remote_resource_h resource, iotcon_error_e err, int
sequence_number, iotcon_response_h response, void *user_data)
{
    ...
    iotcon_response_get_result(response, &response_result);
    if (IOTCON_RESPONSE_OK != response_result) {
        _E("_on_response_observe Response error(%d)", response_result);
        return;
    }

    iotcon_response_get_representation(response, &repr);
    iotcon_representation_get_attributes(repr, &attributes);
    iotcon_attributes_get_bool(attributes, "opened", &opened);

    resource_info->cb_info->cb(resource_info, (void *)(int) opened, resource_info-
>cb_info->user_data);
}
```

Monitoring

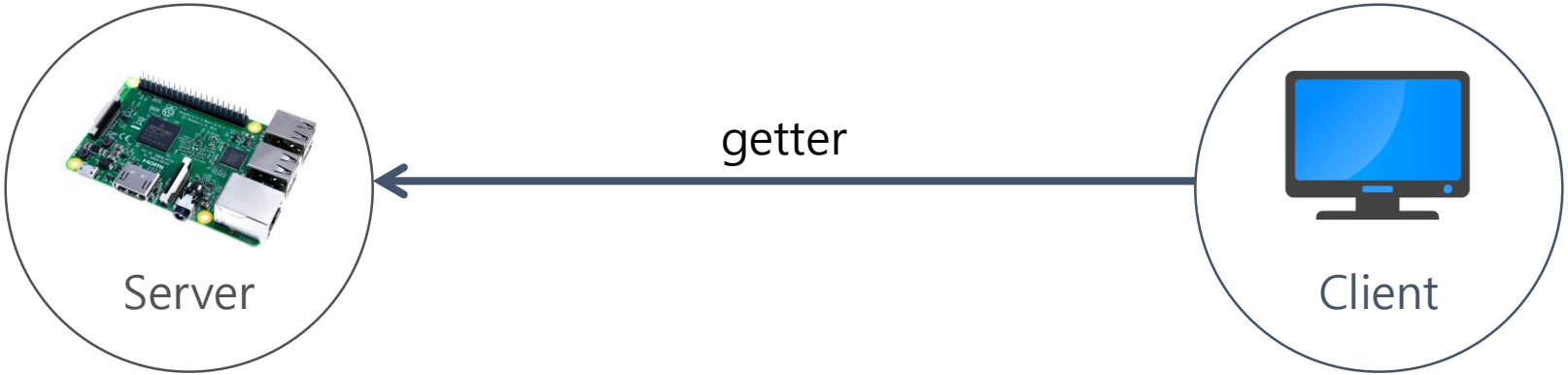



```
int   iotcon_remote_resource_start_monitoring (iotcon_remote_resource_h resource,
          iotcon_remote_resource_state_changed_cb cb,
          void *user_data)

int   iotcon_remote_resource_stop_monitoring (iotcon_remote_resource_h resource)
```

Client

REMOTE RESOURCE

Information of remote resource

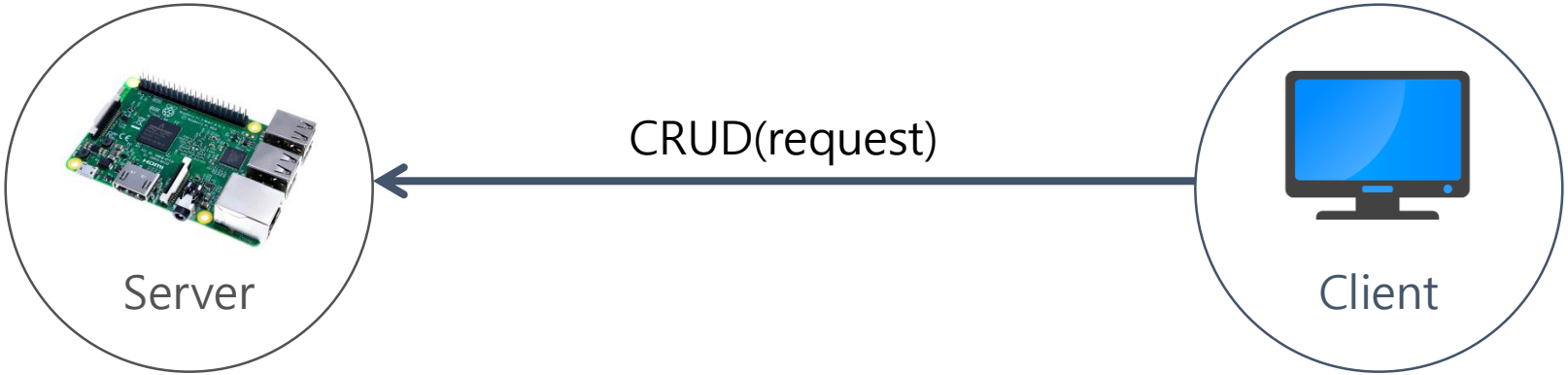


https://goo.gl/8HjgKf

Information of remote resource

```
int iotcon_remote_resource_get_uri_path (iotcon_remote_resource_h resource, char **uri_path)
int iotcon_remote_resource_get_connectivity_type (iotcon_remote_resource_h resource,
        iotcon_connectivity_type_e *connectivity_type)
int iotcon_remote_resource_get_host_address (iotcon_remote_resource_h resource, char **host_address)
int iotcon_remote_resource_get_device_id (iotcon_remote_resource_h resource, char **device_id)
int iotcon_remote_resource_get_device_name (iotcon_remote_resource_h resource, char **device_name)
int iotcon_remote_resource_get_types (iotcon_remote_resource_h resource, iotcon_resource_types_h *types)
int iotcon_remote_resource_get_interfaces (iotcon_remote_resource_h resource,
        iotcon_resource_interfaces_h *ifaces)
int iotcon_remote_resource_get_policies (iotcon_remote_resource_h resource, uint8_t *policies)

int iotcon_remote_resource_get_options (iotcon_remote_resource_h resource, iotcon_options_h *options)
int iotcon_remote_resource_set_options (iotcon_remote_resource_h resource, iotcon_options_h options)
```

CRUD remote resources



CRUD remote resources(client)

```
int iotcon_remote_resource_get (iotcon_remote_resource_h resource,
        iotcon_query_h query, void *user_data)

int iotcon_remote_resource_put (iotcon_remote_resource_h resource,
        iotcon_representation_h repr, iotcon_query_h query,
        iotcon_remote_resource_response_cb cb, void *user_data)

int iotcon_remote_resource_post (iotcon_remote_resource_h resource,
        iotcon_representation_h repr, iotcon_query_h query,
        iotcon_remote_resource_response_cb cb,
        void *user_data)

int iotcon_remote_resource_delete (iotcon_remote_resource_h resource,
        iotcon_remote_resource_response_cb cb, void *user_data)
```

https://goo.gl/8HjgKf

TIZEN

CRUD remote resources(server)

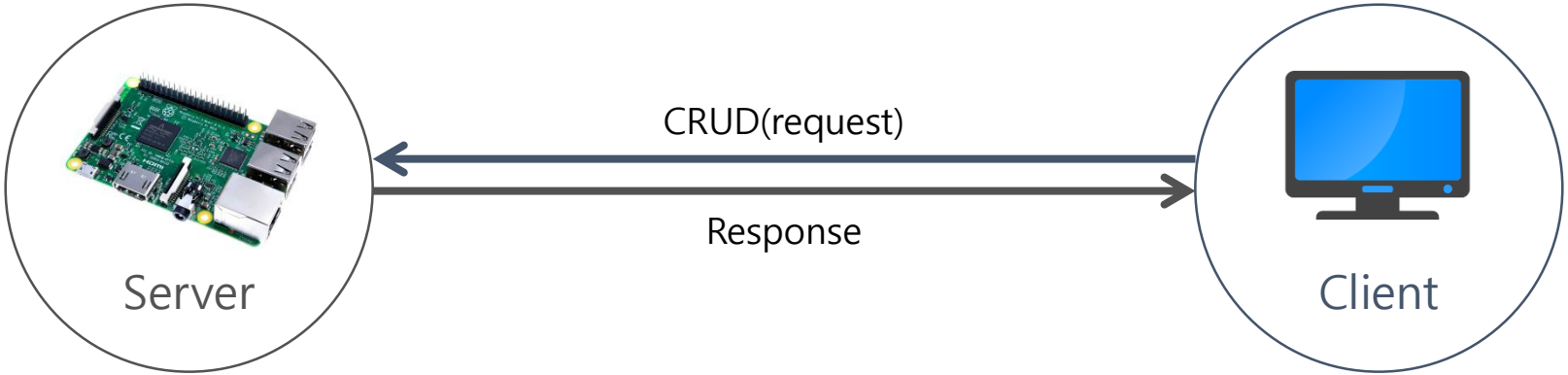
```
int iotcon_resource_create (const char *uri_path,
        iotcon_resource_types_h res_types,
        iotcon_resource_interfaces_h ifaces,
        uint8_t policies,
        iotcon_request_handler_cb cb,
        void * user_data,
        iotcon_resource_h *resource_handle)

int  iotcon_request_get_request_type (iotcon_request_h request, iotcon_request_type_e *type)
```

IOTCON_REQUEST_UNKNOWN
IOTCON_REQUEST_GET
IOTCON_REQUEST_PUT
IOTCON_REQUEST_POST
IOTCON_REQUEST_DELETE

Tizen Talks

TIZEN

Response



https://goo.gl/jc8SVV

TIZEN

Response

```
int  iotcon_response_create (iotcon_request_h request, iotcon_response_h *response)
int  iotcon_response_destroy (iotcon_response_h resp)

int  iotcon_response_get_options (iotcon_response_h resp, iotcon_options_h *options)
int  iotcon_response_get_representation (iotcon_response_h resp, iotcon_representation_h *repr)
int  iotcon_response_get_result (iotcon_response_h resp, iotcon_response_result_e *result)

int  iotcon_response_set_result (iotcon_response_h resp, iotcon_response_result_e result)
int  iotcon_response_set_representation (iotcon_response_h resp, iotcon_representation_h repr)
int  iotcon_response_set_options (iotcon_response_h resp, iotcon_options_h options)

int  iotcon_response_send (iotcon_response_h resp)
```

server

```
static iotcon_representation_h _request_handler_get(door_resource_s *door, iotcon_request_h request)
{
    iotcon_attributes_create(&attributes);
    iotcon_attributes_add_bool(attributes, "opened", door->attributes);

    iotcon_representation_create(&repr);
    iotcon_representation_set_uri_path(repr, door->uri_path);
    iotcon_representation_set_attributes(repr, attributes);

    iotcon_attributes_destroy(attributes);

    iotcon_response_create(request, &response);
    iotcon_response_set_result(response, IOTCON_RESPONSE_OK);
    iotcon_response_set_representation(response, repr);

    iotcon_response_send(response);

    iotcon_response_destroy(response);
    iotcon_representation_destroy(resp_repr);
}
```

client

```
static void _response_get_query(iotcon_remote_resource_h resource, iotcon_response_h response, void *user_data)
{
    ...
    ret = iotcon_response_get_result(response, &response_result);
    ret_if(IOTCON_ERROR_NONE != ret);

    if (IOTCON_RESPONSE_OK != response_result) {
        _E("_response_get_query response error(%d)", response_result);
        return;
    }

    iotcon_remote_resource_get_host_address(resource, &resource_host);
    _I("Resource host : %s", resource_host);

    ret = iotcon_response_get_representation(response, &recv_repr);
    ret_if(IOTCON_ERROR_NONE != ret);

    ret = iotcon_representation_get_attributes(recv_repr, &recv_attributes);
    ret_if(IOTCON_ERROR_NONE != ret);

    ret = iotcon_attributes_get_bool(recv_attributes, "opened", &opened);
    ret_if(IOTCON_ERROR_NONE != ret);
    ...
}
```

TIZEN™

Thank you

Tizen Talks

TIZEN