How To Use the Vuforia Web Services

The Vuforia Web Services enables you to query, upload, and manage images in Cloud Databases and to obtain reports on your targets and databases using a REST based API via HTTP.

The following is an index of articles that explain how to use the VWS API to perform various types of operations.

Cloud Database: Client Access Keys or Server Access Keys?

For managing Cloud DB targets using the VWS API, you will need to use the Server Access Keys associated to your Cloud DB.

Scene setup

In order to access VWS API commands, create a gameobject in your scene with VWS component attached.

VWS component public field AccessKey and SecretKey must be specified for successful authentication.

How To Interpret VWS Result Codes

This table lists the supported result_code, its corresponding HTTP status code, and description.

result_code	Description
“Success”	Transaction succeeded
“TargetCreated”	Target created (target POST response)
“AuthenticationFailure”	Signature authentication failed
“RequestTimeTooSkewed”	Request timestamp outside allowed range
“TargetNameExist”	The corresponding target name already exists (target POST/PUT response)
“UnknownTarget”	The specified target ID does not exist (target PUT/GET/DELETE response)

“BadImage”	Image corrupted or format not supported (target POST/PUT response)
“ImageTooLarge”	Target metadata size exceeds maximum limit (target POST/PUT response)
“MetadataTooLarge”	Image size exceeds maximum limit (target POST/PUT response)
“DateRangeError”	Start date is after the end date
“Fail”	The server encountered an internal error; please retry the request

How To Add a Target

You can add targets to a Cloud Database using the Vuforia Web Services REST API, by executing VWS.AddTarget method. For example,

```
public void AddTarget ()
{
    VWS.Instance.AddTarget(name,
        width,
        image,
        active_flag,
        application_metadata,
        response =>
        {
            if (response.result_code == "Success" || response.result_code == "TargetCreated")
            {
                LogMessage(response.target_id);
            }
            else
```

```
        {
            LogMessage(response.result_code);
        }
      }
    );
  }
```

Field name	Type	Mandatory	Description
name	String [1 - 64]	Yes	Name of the target, unique within a database
width	Float	Yes	Width of the target in scene unit
image	Base64 encoded binary image file in JPG or PNG format	No	Contains the base64 encoded binary recognition image data
active_flag	Boolean	No	Indicates whether or not the target is active for query
application_metadata	Base64 encoded data	No	The base64 encoded application metadata associated with the target

How To Update a Target

Example,

```
  public void UpdateTarget ()
  {
    VWS.Instance.UpdateTarget(target_id,
      name,
      width,
```

```
        image,
        active_flag,
        application_metadata,
        response =>
        {
          if (response.result_code == "Success")
          {
            LogMessage("Target updated");
          }
          else
          {
            LogMessage(response.result_code);
          }
        }
    );
}
```

Field name	Type	Mandatory	Description
name	String [1 - 64]	No	Name of the target, unique within a database
width	Float	No	Width of the target in scene unit
image	Base 64 encoded binary image file in JPG or PNG format	No	Contains the base 64 encoded binary recognition image data
active_flag	Boolean	No	Indicates whether or not the target is active for query

application_metadata	Base 64 encoded data	No	The base 64 encoded application metadata associated with the target

How To Update Target Name

Example,

```
    public void UpdateTargetName ()
{
    VWS.Instance.UpdateTargetName(target_id,
      name,
      response =>
      {
        if (response.result_code == "Success")
        {
          LogMessage("Target name updated");
        }
        else
        {
          LogMessage(response.result_code);
        }
      }
    );
}
```

How To Update Target Width

Example,

```
    public void UpdateTargetWidth ()
{
```

```
    VWS.Instance.UpdateTargetWidth(target_id,
      width,
      response =>
      {
        if (response.result_code == "Success")
        {
          LogMessage("Target width updated");
        }
        else
        {
          LogMessage(response.result_code);
        }
      }
    );
  }
```

How To Update Target Flag

Example,

```
    public void UpdateTargetFlag ()
  {
    VWS.Instance.UpdateTargetFlag(target_id,
      true,
      response =>
      {
        if (response.result_code == "Success")
        {
          LogMessage("Target flag updated");
        }
```

```
            else
            {
               LogMessage(response.result_code);
            }
         }
      );
   }
```

How To Update Target Metadata

Example,

```
   public void UpdateTargetMetadata ()
{
   VWS.Instance.UpdateTargetMetadata(target_id,
      metadata,
      response =>
      {
         if (response.result_code == "Success")
         {
            LogMessage("Target metadata updated");
         }
         else
         {
            LogMessage(response.result_code);
         }
      }
   );
}
```

How To Update Target Image

Example,

```
    public void UpdateTargetImage ()
  {
    VWS.Instance.UpdateTargetImage(target_id,
      image,
      response =>
      {
        if (response.result_code == "Success")
        {
          LogMessage("Target image updated");
        }
        else
        {
          LogMessage(response.result_code);
        }
      }
    );
  }
```

How To Delete a Target

Example,

```
  public void DeleteTarget ()
  {
    VWS.Instance.DeleteTarget(target_id,
      response =>
      {
        if (response.result_code == "Success")
        {
```

```
            LogMessage("Target deleted");
        }
        else
        {
            LogMessage(response.result_code);
        }
    }
  );
}
```

How To Retrieve a Target Record

Example,

```
public void RetrieveTarget()
{
  VWS.Instance.RetrieveTarget(target_id, response =>
    {
      if (response.result_code == "Success")
      {
        string log = "Name: " + response.target_record.name + "\n";
        log += "Width: " + response.target_record.width + "\n";
        log += "Active: " + response.target_record.active_flag + "\n";
        log += "Rating: " + response.target_record.tracking_rating;
        LogMessage(log);
      }
      else
      {
        LogMessage(response.result_code);
      }
```

```
        }
    );
}
```

Field name	Type	Mandatory	Description
result_code	String [1 - 64]	Yes	One of the VWS API Result Codes
transaction_id	32-character String (UUID)	Yes	ID of the transaction
target_record	targetRecord	Yes	Contains a target record at the TMS
status	String [1-64]	Yes	Status of the target; current supported values are “processing,” “success,” and “failed”

Target record body

The target_record body is defined as listed in the following table.

Field name	Type	Mandatory	Description
target_id	32-character String (UUID)	Yes	Target_id of the target
active_flag	Boolean	No	Indicates whether or not the target is active for query; the default is true
name	String [1-64]	Yes	Name of the target; unique within a database
width	Float	Yes	Width of the target in scene unit
tracking_rating	Int [0 - 5]	Yes	Rating of the target recognition image for tracking purposes

reco_rating	string	No	Rating of the target recognition image for recognition purposes – an empty string for now

How To Check for Duplicate Targets

Example,

```
public void RetrieveTargetDuplicates()
{
    VWS.Instance.RetrieveTargetDuplicates(target_id, response =>
        {
            if (response.result_code == "Success")
            {
                LogMessage(response.similar_targets);
            }
            else
            {
                LogMessage(response.result_code);
            }
        }
    );
}
```

Field Name	Type	Mandatory	Description
similar_targets	targetID [0..16]	Yes	List of possible duplicate targets – target_id's for duplicate targets

How To Get a Target List for a Cloud Database

Example,

```
public void LoadTargetList ()
{
    VWS.Instance.RetrieveTargetList( response =>
    {
      if (response.result_code == "Success")
      {
        LogMessage(response.results);
      }
      else
      {
        LogMessage(response.result_code);
      }
    }
  );
}
```

Field	Type	Mandatory	Description
result_code	String [1 - 64]	Yes	One of the VWS API Result Codes
transaction_id	32-character String (UUID)	Yes	ID of the transaction
results	target_id [0..unbounded]	Yes	List of target IDs for the target in the developer project

How To Retrieve a Target Summary Report

Example,

```
public void RetrieveTargetSummary()
{
```

```
    VWS.Instance.RetrieveTargetSummary(target_id, response =>
      {
        if (response.result_code == "Success")
        {
          string log = "DB Name: " + response.database_name + "\n";
          log += "Name: " + response.target_name + "\n";
          log += "Date: " + response.upload_date + "\n";
          log += "Status: " + response.status + "\n";
          log += "Total Recos: " + response.total_recos + "\n";
          log += "Current Month: " + response.current_month_recos + "\n";
          log += "Previous Month: " + response.previous_month_recos;

          LogMessage(log);
        }
        else
        {
          LogMessage(response.result_code);
        }
      }
    );
  }
```

Field name	Type	Mandatory	Description
result_code	String [1 - 64]	Yes	One of the VWS API Result Codes
transaction_id	32-character String (UUID)	Yes	ID of the transaction

database_name	String [1 - 64]	Yes	Name of the database
target_name	String [1 - 64]	Yes	Name of the target
upload_date	Date	Yes	Date of the upload (specified as YYYY-MM-DD)
active_flag	Boolean	Yes	Indicates whether or not the target is active for query; default is true
status	String [1- 64]	Yes	Status of the target; current supported values are processing, success, and failure
tracking_rating	Int [0 - 5]	No*	Rating of the target recognition image for tracking purposes
reco_rating	String	No*	Currently an empty string
total_recos	Int	No*	
current_month_recos	Int	No*	
previous_month_recos	Int	No*	

How To Get a Database Summary Report

Example,

```
public void ConnectToDatabase ()
{
  VWS.Instance.RetrieveDatabaseSummary( response =>
    {
      if (response.result_code == "Success")
```

```
        {
            string log = "Name: " + response.name + "\n";
            log += "Active images: " + response.active_images + "\n";
            log += "Failed images: " + response.failed_images + "\n";
            log += "Inactive images: " + response.inactive_images;

            LogMessage(log);
        }
        else
        {
            LogMessage(response.result_code);
        }
    }
  );
}
```

Field name	Type	Mandatory	Description
result_code	String [1 - 64]	Yes	One of the VWS API Result Codes
transaction_id	32-character String (UUID)	Yes	ID of the transaction
name	String [1 - 64]	Yes	Name of the database
active_images	uint32	Yes	Total number of images with active_flag = true, and status = success for the database
inactive_images	uint32	Yes	Total number of images with active_flag = false, and status = success for the database

failed_images	uint32	Yes	Total number of images with status = fail for the data