

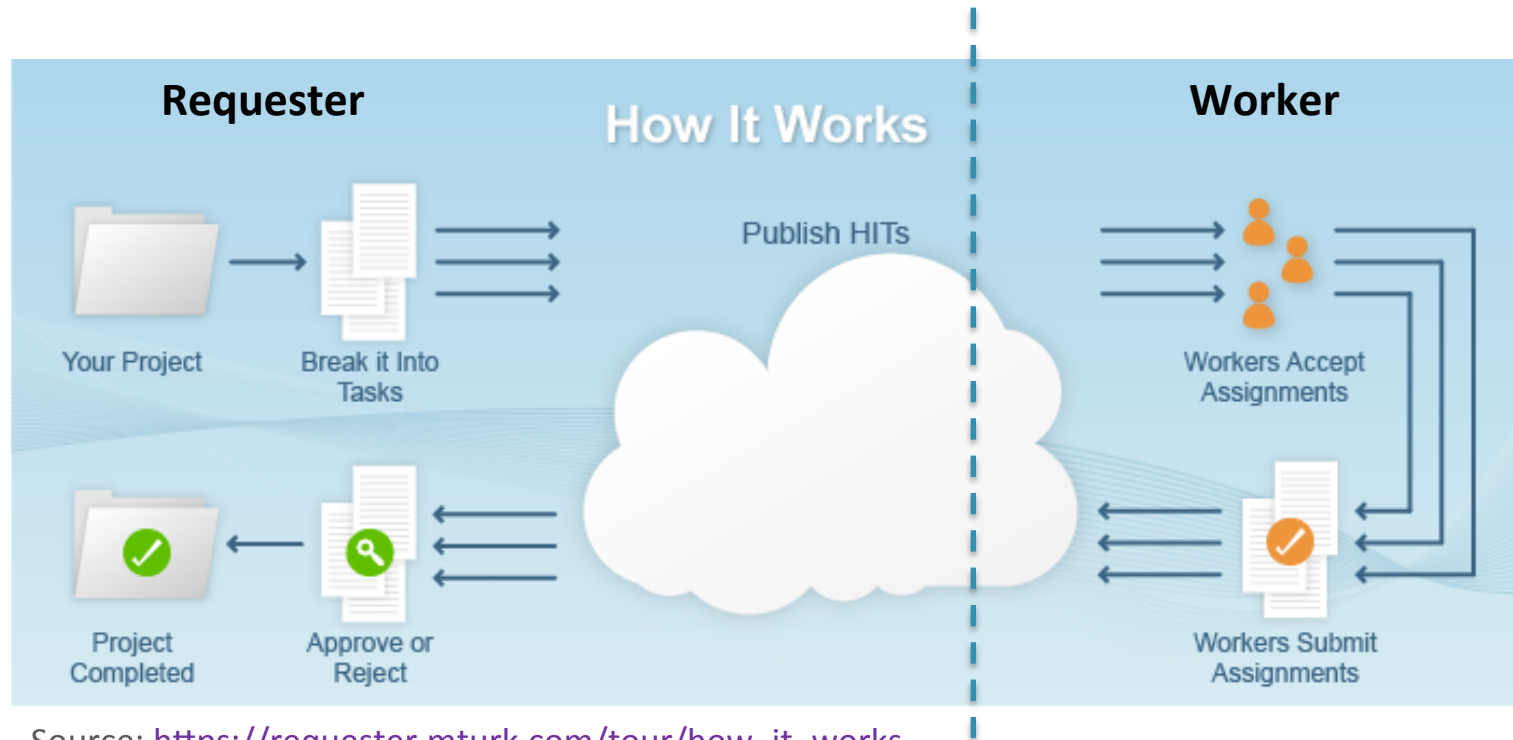
Amazon Mechanical Turk **Hands-on session**

Maribel Acosta





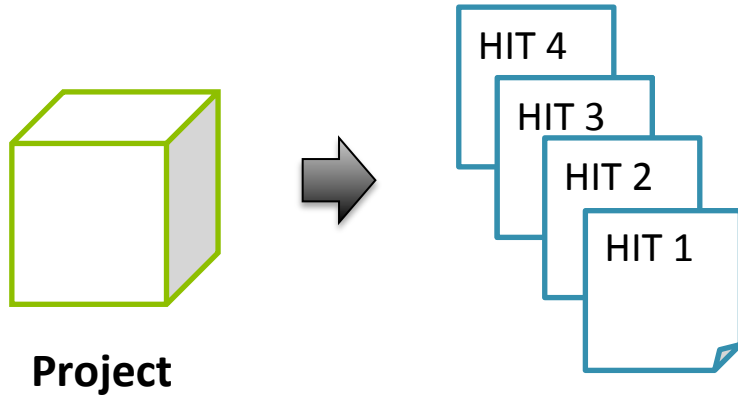
MTurk Basic Concepts (1)



Source: https://requester.mturk.com/tour/how_it_works

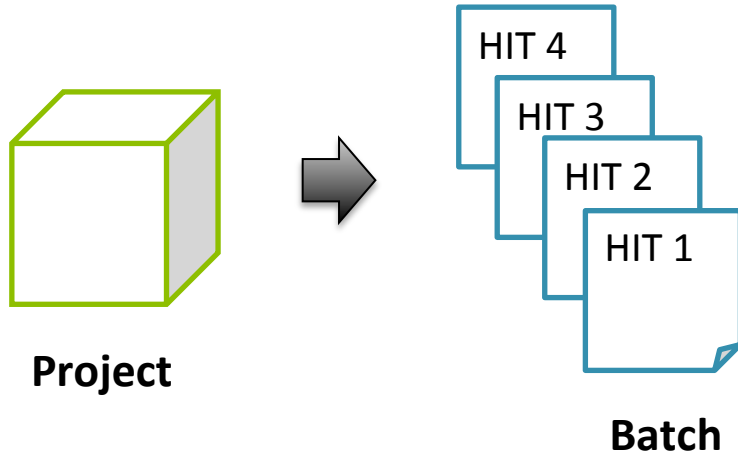
- **Requester:** creates and submits tasks to the platform.
- **Worker:** person who solves the tasks.
- **Human Intelligence Task (HIT):** work unit.

MTurk Basic Concepts (2)



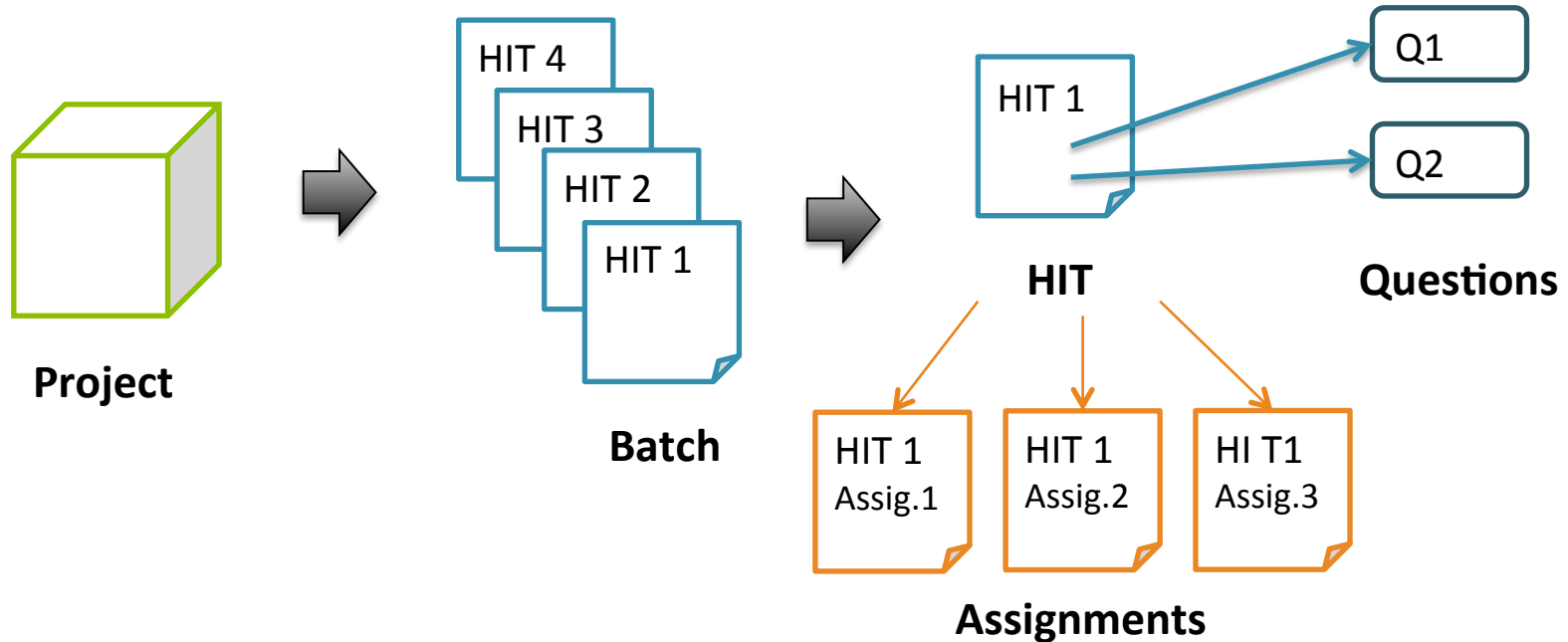
- **Project:** HIT HTML + HIT metadata
 - The elements that stay the same in every HIT are denominated **template**
 - The data that will vary from HIT to HIT are specified via **variables**
- NOTE: If no variables are specified in the project, we will create a single HIT
- **Variables:** allow creating several HITs in the project

MTurk Basic Concepts (3)



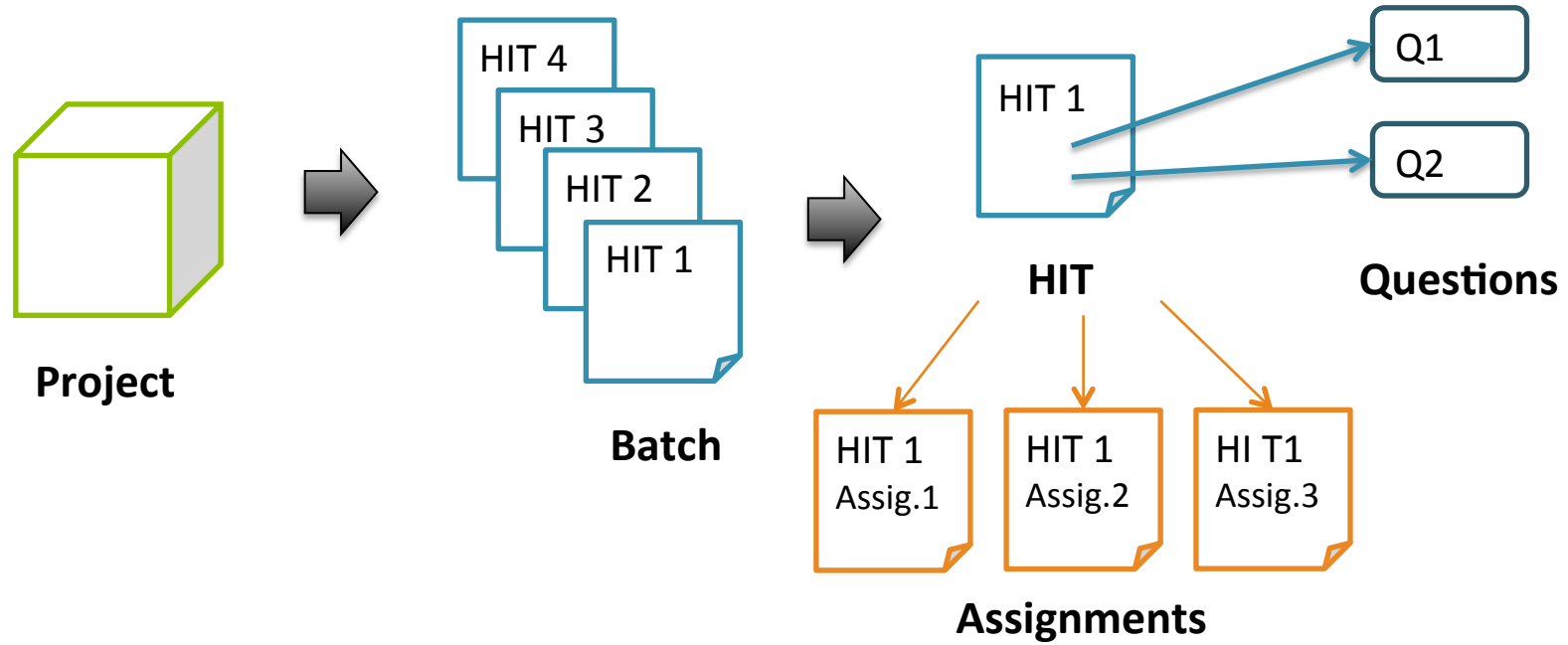
- **Batch:** Group of HITs created by instantiating the variable(s) of a project
- The values of the variables are specified in (CSV, TSV) files:
 - Each **column** corresponds to a variable
 - Each **row** is an instance -> HIT
 - Each file corresponds to a batch
- We can create **several batches** for the same project

MTurk Basic Concepts (4)



- **HIT:** Work unit. The same HIT can be solved by 1 or more workers (assignments)
- **Assignment:** How many workers should solve one exact same HIT
- **Questions:** A single HIT may contain one or several questions

MTurk Basic Concepts (5)

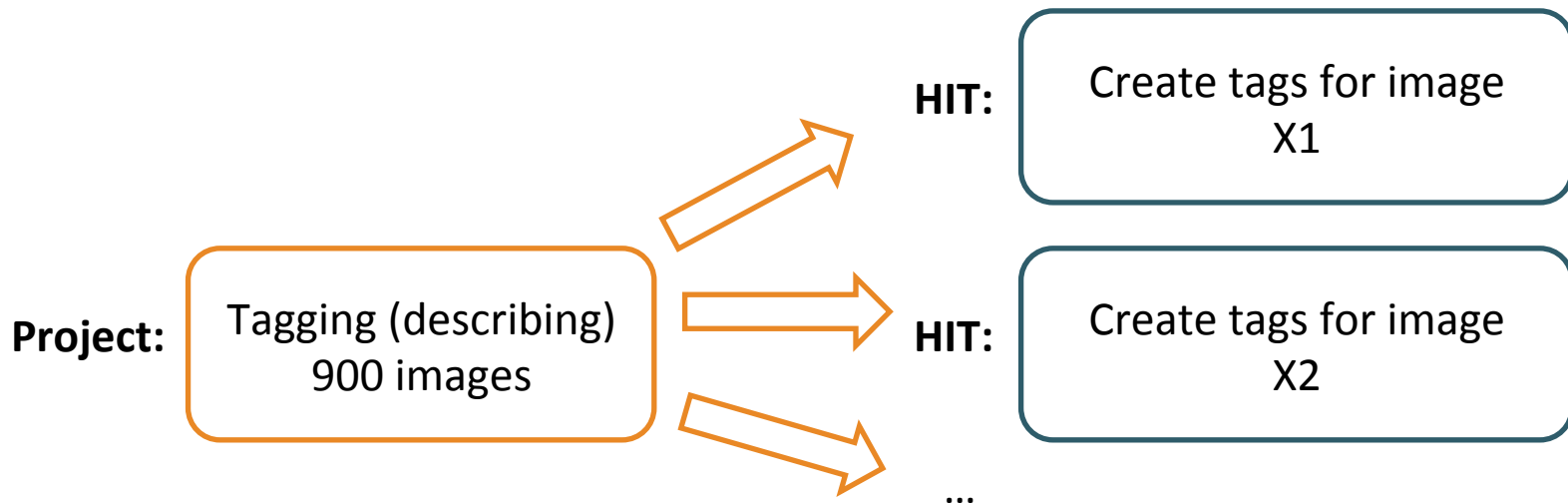


Total cost of the project = No. of HITs x No. of Assignments x (Reward per HIT + Fee)

MTurk Basic Concepts (6)

Example of Human Intelligence Tasks (HITs)

- Projects can be broken into smaller tasks called HITs
- A HIT represents a single work unit

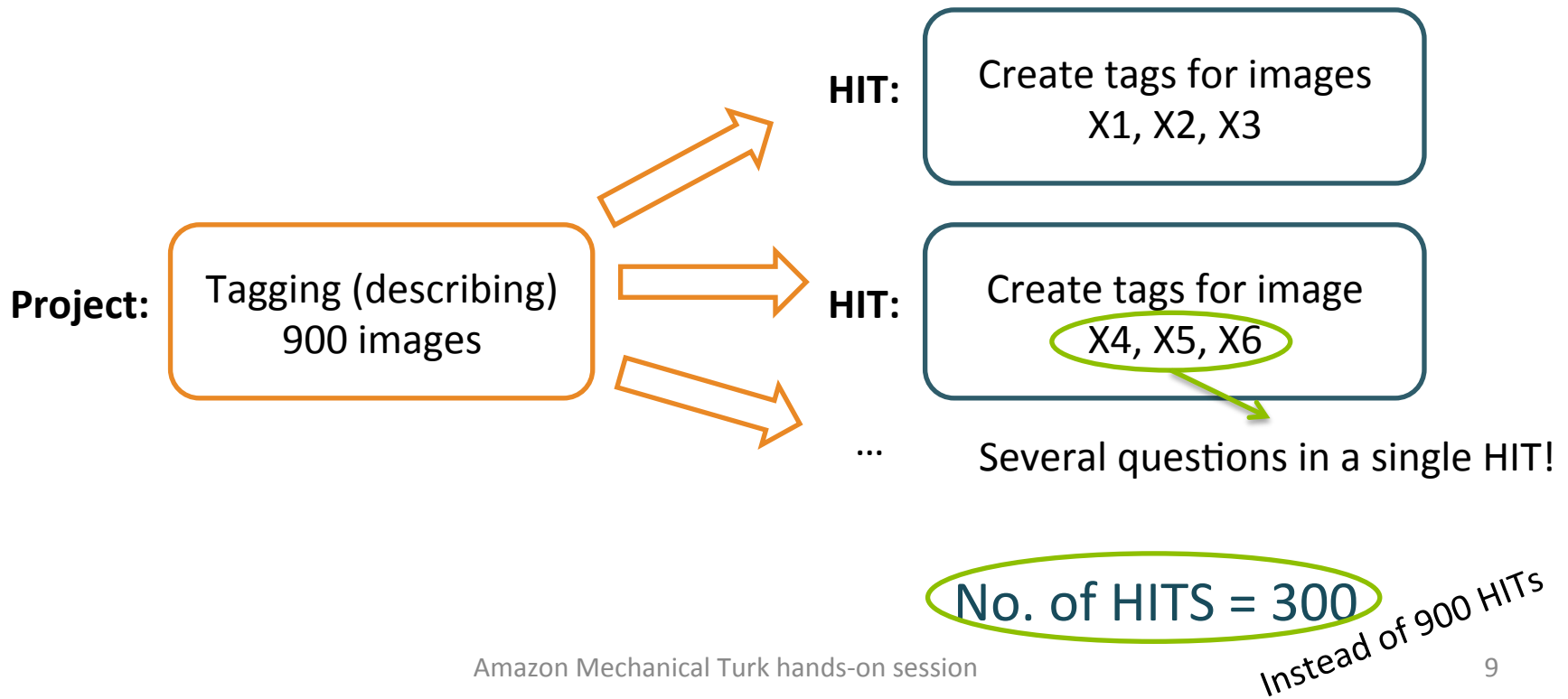


No. of HITS = 900

MTurk Basic Concepts (7)

Example of Human Intelligence Tasks (HITs)

- Projects can be broken into smaller tasks called HITs
- A HIT represents a single work unit

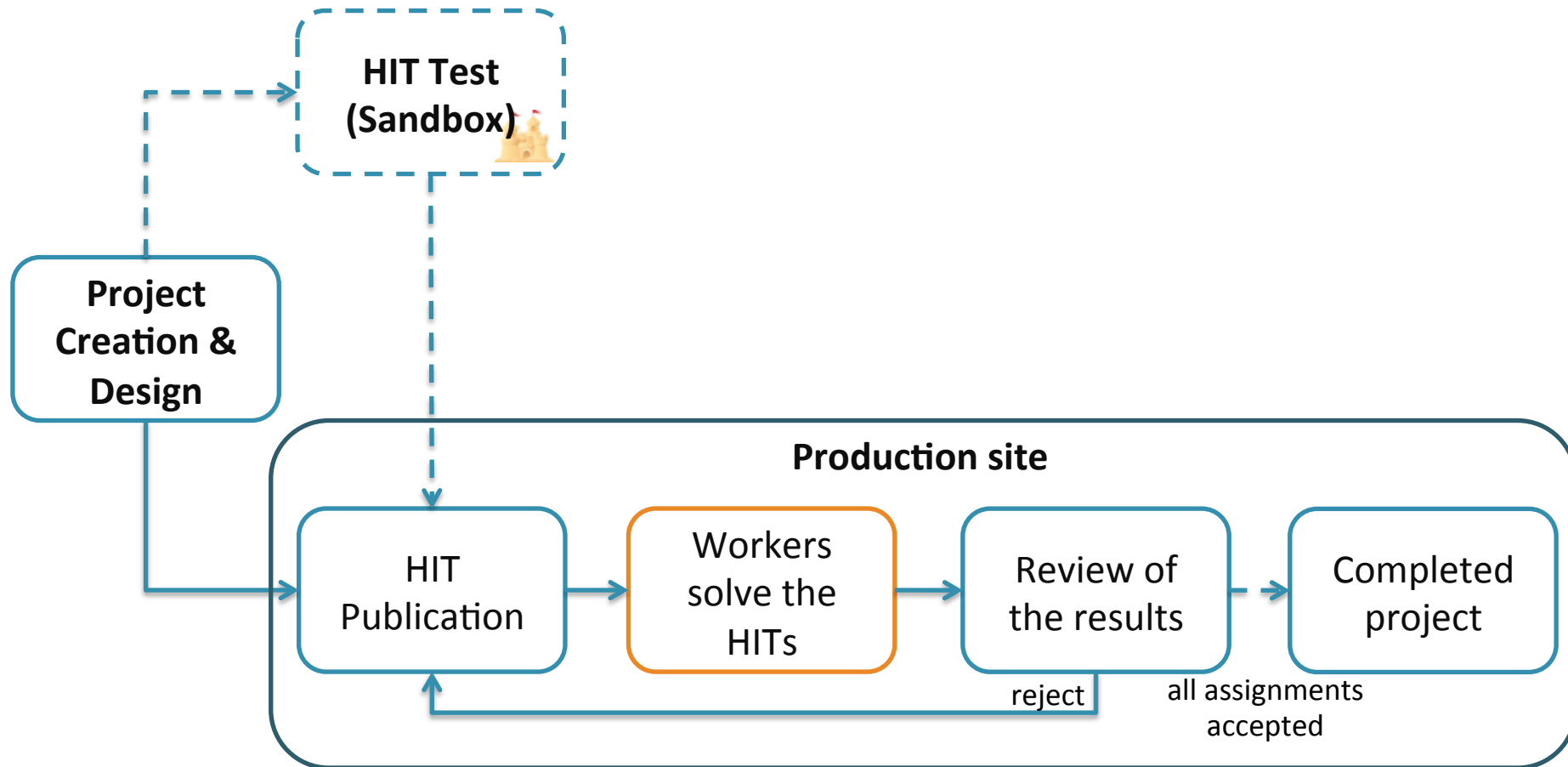


MTurk Basic Concepts (8)

When creating a project or individual HITs, the **HIT properties** must be specified:

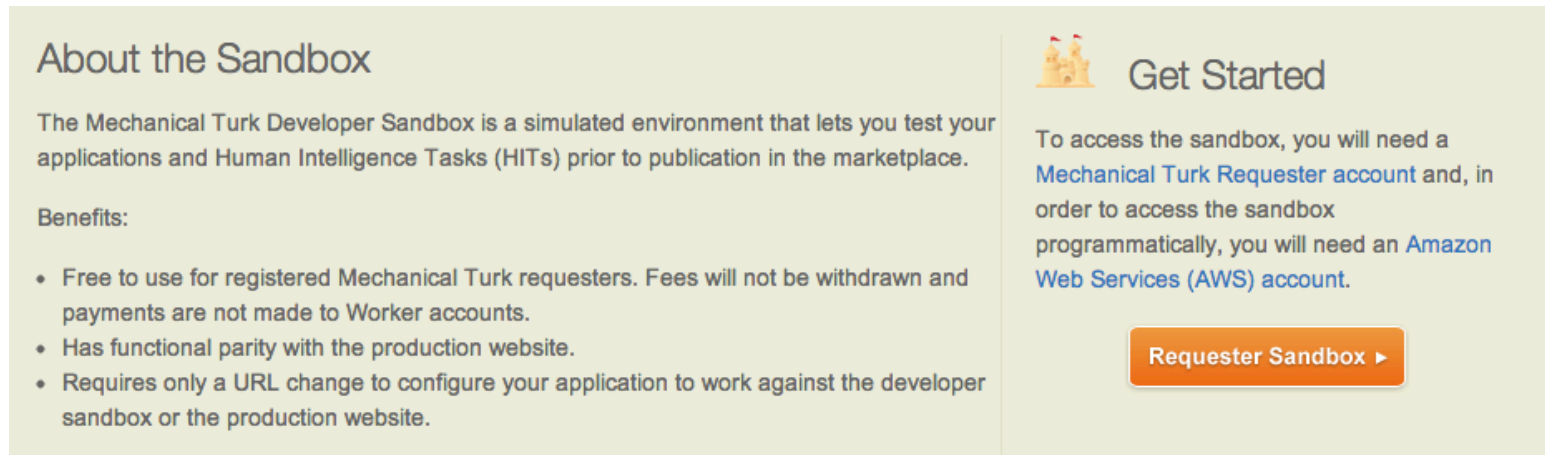
- **General information:** includes the title and description of the HIT, as well as keywords which are used by worker for searching HITs
- **HIT duration time:** time allotted to solve the HIT (before it is given to another worker)
- **HIT life time:** how long will the HIT be available on the platform
- **# Assignments:** number of different persons that will perform the exact same HIT
- **Reward:** payment for correctly solving each assignment

MTurk Workflow for Requesters



MTurk Sandbox

The Sandbox is a simulated MTurk environment to test HITs.

A screenshot of the MTurk Developer Sandbox page. The page is divided into two main sections: 'About the Sandbox' on the left and 'Get Started' on the right. The 'About the Sandbox' section describes the environment and lists three benefits: it's free for registered requesters, has functional parity with the production website, and requires only a URL change. The 'Get Started' section includes a castle icon, explains that a Mechanical Turk Requester account and an Amazon Web Services (AWS) account are needed for programmatic access, and features an orange button labeled 'Requester Sandbox ►'.

- Log in as **requester**: preview and test the interface of your HITs
 - <https://requestersandbox.mturk.com>
- Log in as **worker**: solve your own HITs to test their functionalities and result output
 - <https://workersandbox.mturk.com>
- **Best practice**: Always test your HITs (**as requester and worker**) before publishing them in the production site

Managing HITs in MTurk

There are three different mechanism to manage your HITs in MTurk:

Web
Interface



Command
Line Tools



API



MTURK WEB INTERFACE

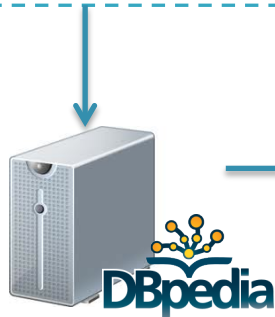
Hands On!

- **Project:** Crowdsourcing DBpedia triples to verify the links to external web pages

```
prefix dbpedia-ont:<http://dbpedia.org/ontology/>
prefix foaf:<http://xmlns.com/foaf/0.1/>
SELECT *
WHERE {
  ?s dbpedia-ont:wikiPageExternalLink ?o;
    foaf:name ?s_name;
    foaf:isPrimaryTopicOf ?s_wikipedia .
} LIMIT 200
```

→ Triple to crowdsource

} Triples to build the UI

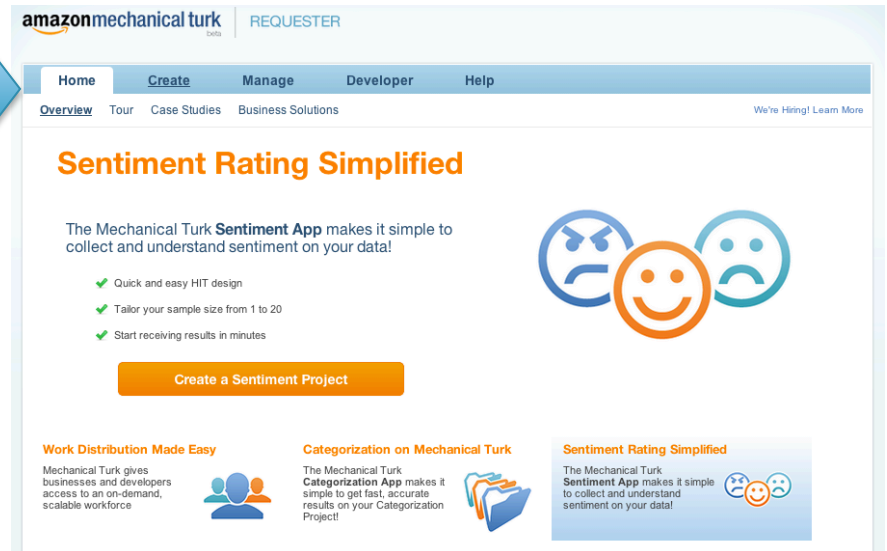


MTurkDemo/data/sparql.csv

<http://dbpedia.org/sparql>

Hands On!

- Go to MTurk Sandbox as a **requester**:
 - <https://requestersandbox.mturk.com/>
- Click on **Sign In**
 - Email address
 - Password
- Now we are at “home”



1. Creating a Project

The screenshot displays the Amazon Mechanical Turk requester dashboard. At the top, the 'Create' button in the navigation bar is highlighted with a red rectangle. Below it, the 'New Project' link is also highlighted with a red rectangle. A sidebar on the left, titled 'Start a New Project', lists various project templates, with the entire list enclosed in a red rounded rectangle. The 'Example of Categorization' section on the right shows a task where the user must select the best category for a given image of a bedroom. The categories are kitchen, living, bath, bed, and outside. A note at the bottom of this section states: 'You must ACCEPT the HIT before you can submit the results.'

amazonmechanical turk beta REQUESTER

Home **Create** Manage Developer Help

New Project New Batch with an Existing Project Create HITs individually


Start a New Project

Categorization

- Data Collection
- Moderation of an Image
- Sentiment
- Survey
- Survey Link
- Tagging of an Image
- Transcription from A/V
- Transcription from an Image
- Writing
- Other

Example of Categorization

Choose the best category for this image



[View Instructions ↓](#)

Select the room location in home for this picture. Seating areas outside are outside not living. Offices or dens are living not bedrooms. Bedrooms should contain a bed in the picture.

- ☐ kitchen
- ☐ living
- ☐ bath
- ☐ bed
- ☐ outside

You must ACCEPT the HIT before you can submit the results.

Different predefined templates: Select “other”

2. Setting up the HIT Properties (1)

amazonmechanical turk beta REQUESTER

Home Create Manage Developer Help

New Project New Batch with an Existing Project [Create HITs individually](#)

Edit Project

Specify the properties that are common for all of the HITs created using this project.

1 Enter Properties 2 Design Layout 3 Preview and Finish

Project Name: This name is not displayed to Workers.

Describe your HIT to Workers

Title
Describe the task to Workers. Be as specific as possible, e.g. "answer a survey about movies", instead of "short survey", so Workers know what to expect.

Description
Give more detail about this task. This gives Workers a bit more information before they decide to view your HIT.

Keywords
Provide keywords that will help Workers search for your HITs.

☐ This project may contain potentially explicit or offensive content, for example, nudity. ([See details](#))

HIT description

2. Setting up the HIT Properties (2)

Setting up your HIT

Reward per assignment
Tip: Consider how long it will take a Worker to complete each task. A 30 second task that pays \$0.05 is a \$6.00 hourly wage.

Number of assignments per HIT
How many unique Workers do you want to work on each HIT?

Time allotted per assignment **Minutes**
Maximum time a Worker has to work on a single task. Be generous so that Workers are not rushed.

HIT expires in **Days**
Maximum time your HIT will be available to Workers on Mechanical Turk.

Results are automatically approved in **Days**
After this time, all unreviewed work is approved and Workers are paid.

HIT properties

Very IMPORTANT:
Set up quality mechanisms
Masters are selected by default

[Advanced »](#)

3. Selecting Qualifications

Advanced

Worker requirements «

Worker requirements:

Customize Worker Requirements...

Specify ALL the qualifications Workers must meet to work on your HITs:

Masters remove

HIT Approval Rate (%) greater than or equal to 95 remove

Number of HITs Approved greater than or equal to 1000 remove

✓ -- Select -- to 0 remove

Masters to 0 remove

Categorization Masters

Photo Moderation Masters

Masters

System Qualifications

HIT submission rate (%)

Location

HIT rejection rate (%)

HIT Approval Rate (%)

Number of HITs Approved

Only Y

view my HITs.

Worker requirements (filters)

Masters expect higher rewards
MTurk charges 20% for masters

4. Defining the Task

Edit Project

Use the HTML editor below to design the layout of your HIT. This layout is common for all of the HITs created with this project. You can define variables for data that will vary from HIT to HIT ([Learn more](#)).

1 Enter Properties

2 Design Layout

3 Preview and Finish





Project Name: This name is not displayed to Workers.



Frame Height Height in pixels of the frame your HIT will be displayed in to Workers. Adjust the height appropriately to minimize scrolling for Workers.



Format


Font

U *I* **B** A *I*_x

 Source

Instructions

[Jump to questions](#)

In this task, you will help us verify whether the links to external pages contained in Wikipedia articles are correct or not. The content of these external pages should be related to the content of the Wikipedia article. In this task, you will verify whether this is the case or not. In order to solve this task, we will provide the Wikipedia article and an external website that the article links to.

Your job: Compare whether the Wikipedia article and the external website are related.
Try to **refresh the page** if the content is not displayed properly.

Example of **incorrect** data

In the following example, we are checking whether the external web pages are related to the Wikipedia article "John Two-Hawks".

body h2

WYSIWYG HTML editor



4. Defining the Task (with Variables)

[Edit Project](#)

Use the HTML editor below to design the layout of your HIT. This layout is common for all of the HITs created with this project. You can define variables for data that will vary from HIT to HIT ([Learn more](#)).

The screenshot shows the Hitlens interface with three tabs: 'Enter Properties', 'Design Layout', and 'Preview and Finish'. The 'Design Layout' tab is active. The 'Project Name' is 'DBpedia outlinks'. The 'Frame Height' is set to 400. The 'Question' is 'The content in "External page" corresponds to the topic covered in the Wikipedia article?'. Below the question, the text 'About:' is followed by a red box containing the variable placeholder `${s_name}`. Below that, the text 'Wikipedia article about: `${s_name}`' is also enclosed in a red box. Red arrows point from these two boxes to the 'Variables' definition on the right.

Template: elements that stay the same in every HIT

Variables: data that will vary from HIT to HIT. Are denoted as follows: `${var_name}`

5. Previewing the Template

Edit Project

This is how your HIT will look to Mechanical Turk Workers. Before you publish these HITs, any variables in the HIT will be replaced with the input data that you provide when you publish the HIT. You can download a sample of the input file for this project or learn more about [acceptable file formats](#)

[Download sample](#)

1 Enter Properties

2 Design Layout

3 Preview and Finish

This is what the workers will see

Project Name: DBpedia outlinks

This name is not displayed to Workers.

Comparing content between two web pages

Requester: mac21

Reward: \$0.04 per HIT

HITs available: 0

Duration: 10 Minutes

Qualifications Required: HIT Approval Rate (%) greater than or equal to 95 , Number of HITs Approved greater than or equal to 1000

HIT Preview

Question

The content in "External page" corresponds to the topic covered in the Wikipedia article?

About:

`${s_name}`

[Wikipedia article about: `\${s_name}`](#)

The variables will be replaced by the input data

6. Creating Batches

The screenshot shows the Amazon Mechanical Turk interface for creating a new batch. The top navigation bar includes 'amazonmechanicalturk' and 'REQUESTER'. The left sidebar has 'Home' and 'Create' tabs, with 'New Project' and 'New Batch with an...' links. The main area is titled 'New Batch' and contains the instruction 'Choose a .csv file with the variables you specified in your project.' Below this are three buttons: 'Choose File', '<no file selected>', and 'Upload'. A table below lists existing batches. The first batch, 'DBpedia outlinks', is highlighted with a red box. A red arrow points from the 'New Batch' button in this row to a detailed view of the batch. The detailed view shows the 'Layout ID: 27MSNQQ7Q67066KV8KI15TQC1Z3UAT' and 'Parameters: o, s_name, s_wikibase'. A small window shows the 'sparql.csv' file content.

Project Name	Title	Creation Date			
DBpedia outlinks	Comparing content between two web pages	October 16, 2013	New Batch	Edit	Copy

DBpedia outlinks

Layout ID: 27MSNQQ7Q67066KV8KI15TQC1Z3UAT

Parameters: o, s_name, s_wikibase

sparql.csv

```
1 "s","o","s_name","s_wikibase"
2 "http://dbpedia.org/resource/!Action_Pact!","http://www.
3 "http://dbpedia.org/resource/!Action_Pact!","http://www.
4 "http://dbpedia.org/resource/!Action_Pact!","http://www.
5 "http://dbpedia.org/resource/!Action_Pact!","http://www.
```


7. Previewing the HITs

DBpedia outlinks

Comparing content between two web pages

Requester: mac21 Reward: \$0.04 per HIT HITs available: 200 Duration: 10 Minutes

Qualifications Required: HIT Approval Rate (%) greater than or equal to 95 ,
Number of HITs Approved greater than or equal to 1000

HIT Preview

Question

The content in "External page" corresponds to the topic covered in the Wikipedia article?

About:

!Action Pact!

[Wikipedia article about: !Action Pact!](#)



The screenshot shows a Wikipedia article interface. On the left is a globe icon with various characters. To its right are tabs for 'Article', 'Talk', and 'Read', with 'Article' selected. Below these is a search bar with the text '!Action Pact!'. Above the search bar, the text 'About:' is followed by a red-bordered box containing '!Action Pact!'. To the right of this box, the text 'Variables are replaced with the data from the input file' has two red arrows pointing to the box and the search bar. At the top right of the preview area, there are links for 'Create account' and 'Log in'.

8. Publishing the HITs

DBpedia outlinks

Batch Summary		
Batch Name	HITs	between a Wikipedia article
Batch Product	Number of HITs in this batch:	200
Title:	Number of assignments per HIT:	x 3
Description:	Total number of assignments in this batch:	600
Batch Example		
Results		
Workers		
Qualifications		
HITs		
Number		
Number		
Total number		
Cost		
Reward		
Estimated		
Estimated fees to Mechanical Turk		
Estimated Total Cost:	\$27.000	(this is the amount that will be deducted from your Available Balance when you click "Publish HITs")
Your Available Balance:	\$10,000.000	(before clicking "Publish HITs")
Your Projected Balance:	\$9,973.000	(after clicking "Publish HITs")

Summary of the project:

- # of HITs
- Rewards
- Total payment
- Account balance

9. Retrieving the Results

Review Results

Select the check boxes on the left to approve or reject results. You only pay for approved results. To evaluate results offline, select Download CSV.

For additional batch information, [view batch details](#).

DBpedia outlinks 10

Customize ViewFilter ResultsUpload CSVDownload CSV

1 of 1 assignments (FILTER APPLIED: only show assignments that are in 'Submitted' status)

ApproveReject

<input type="checkbox"/>	HIT ID ▲	Worker ID	Lifetime Approval Rate	Input Data	Q1
<input type="checkbox"/>	2N6Q1SNQQ7Q685ZCMH3NU12T1XJS8I	A3QBKUNGNMLHTF	98% (131/133)	BatchId:58757;	correct
<input type="checkbox"/>	HIT ID ▲	Worker ID	Lifetime Approval Rate	Input Data	Q1

ApproveReject

MTURK COMMAND LINE TOOLS



List of Commands

[HITs]

- Make template
- Load HITs
- Delete HITs
- Update HITs
- Extend HITs

[Results]

- Get result
- Approve/reject work
- Review results
- Generate result summary
- Grant bonus
- Block/unblock worker

[Qualification types]

- Create qualification type
- Update qualification type
- Get qualification results
- Evaluate qualification request
- Approve/reject qualification request
- Revoke qualifications
- Update qualification score

[Account]

- Reset account
- Get balance

Creating HITs with CLT

1. Configuring the connection to the platform
2. Setting up the HIT properties
3. Defining the task
4. Creating the batch
5. Publishing the HITs
6. Retrieving the results

1. Configuring the connection to the platform

- Configure the `./bin/mturk.properties` file
- Add the keys to access your MTurk account

```
# Information to access the MTurk account.  
access_key=  
secret_key=
```

- Specify the service to use (Sandbox or production site)

```
# If you want to use the Sandbox use the following service_url  
service_url=https://mechanicalturk.sandbox.amazonaws.com/?Service=AWSMechanicalTurkRequester  
  
# If you want to use the Sandbox use the following service_url  
#service_url=https://mechanicalturk.amazonaws.com/?Service=AWSMechanicalTurkRequester
```

2. Setting up the HIT Properties

- Create a HIT properties file `myhits.properties`

```
#####
## Basic HIT Properties
#####

title: Comparing content between two pages
description: Compare the content between a Wikipedia article and an external page.
reward:0.04
assignments:3
annotation:Wikipedia#article#DBpedia#data#errors

#####
## HIT Timing Properties
#####

# this Assignment Duration value is calculated based on the number questions.
assignmentduration:900

# this HIT Lifetime value is 60*60*24*15 = 15 days
hitlifetime:1296000

# this Auto Approval period is 60*60*24*15 = 15 days
autoapprovaldelay:1296000

#####
## Qualification Properties
#####

# Qualifications can be defined in the properties file instead of in code.
# You can add multiple qualifications for this HIT by simply increasing the # suffix.
# i.e. qualification.2: XXXXX
#      qualification.comparator.2:greaterthan

# this is a built-in qualification -- user must have an approval rate of 25% or greater
qualification.1:0000000000000000L0
qualification.comparator.1:greaterthan
qualification.value.1:50
qualification.private.1:false
```


3. Defining the Task (1)

- Create a template file myhits.question

Data
structure

```
<?xml version="1.0" encoding="UTF-8"?>
<QuestionForm
  xmlns="http://mechanicalturk.amazonaws.com/AWSMechanicalTurkDataSchemas/2005-10-01/
  QuestionForm.xsd">
  <Overview>
    <FormattedContent><![CDATA[
      <h2>Instructions</h2>

      In this task, you will help us verify whether the links to external pages contained
      in Wikipedia articles are correct or not. The content of these external pages should
      be related to the content of the Wikipedia article. In this task, you will verify
      whether this is the case or not. In order to solve this task, we will provide the
      Wikipedia article and an external website that the article links to.<br />
      <br />
      <b>Your job:</b> Compare whether the Wikipedia article and the external website are
      related.<br />
      Try to <b>refresh the page</b> if the content is not displayed properly.
      <br />

      <h3>Example of <font color="#C92020">incorrect</font> data</h3>

      <p>In the following example, we are checking whether the external web pages are
      related to the Wikipedia article &quot;John Two-Hawks&quot;.<br />
      The following link shows a website not related to &quot;John Two-Hawks&quot;.</p>

      About: <h2>John Two-Hawks</h2>
      <br />
      <b>External page:</b><a href="http://www.cedarlakedvd.com"
      target="_blank">http://www.cedarlakedvd.com</a>

      <iframe height="200" src="http://www.cedarlakedvd.com" width="620">Preview not
      available. Please go to http://www.cedarlakedvd.com.</iframe><br />
    ]]>
  </FormattedContent>
  </Overview>
</QuestionForm>
```

FormattedContent allows
for using HTML tags

Instructions

3. Defining the Task (2)

- Create a template file `myhits.question`

Question content

```
<Question>
  <QuestionIdentifier>1</QuestionIdentifier>
  <QuestionContent>
    <FormattedContent><![CDATA[
      <p><b>The content in &quot;External page&quot;; corresponds to the topic covered in
        the Wikipedia article?</b></p>

      About: <b>${s_name}</b>

      <h3><a href="${s_wikipage}" target="_blank">Wikipedia article about:
        ${s_name}</a></h3>

      <p><iframe height="300" src="${s_wikipage}" width="620">Preview not available.
        Please go to ${s_wikipage}</iframe></p>

      <h3><a href="${o}" target="_blank">External page: ${o}</a></h3>

      <p><iframe height="300" src="${o}" width="620">Preview not available.
        Please go to ${o}</iframe></p>

      <p>Your answer:</p>
    ]]></FormattedContent>
  </QuestionContent>
  <AnswerSpecification>
    <SelectionAnswer>
      <MinSelectionCount>1</MinSelectionCount>
      <MaxSelectionCount>1</MaxSelectionCount>
      <StyleSuggestion>radiobutton</StyleSuggestion>
      <Selections>
        <Selection>
          <SelectionIdentifier>Correct</SelectionIdentifier>
          <Text>Correct</Text>
        </Selection>
        <Selection>
          <SelectionIdentifier>Incorrect</SelectionIdentifier>
          <Text>Incorrect</Text>
        </Selection>
      </Selections>
    </SelectionAnswer>
  </AnswerSpecification>
</Question>
```

Variables

Type of selection

Option 1

Option 2

3. Defining the Task (3)

Question data structures

[QuestionForm]

Describes one or more questions for a HIT, or for a Qualification test.

Elements: QuestionIdentifier, DisplayName, IsRequired, QuestionContent, AnswerSpecification

[ExternalQuestion]

Displays a web page from your website in a frame.

Elements: ExternalURL, FrameHeight

[HTMLQuestion]

Describes the HIT with HTML code, without running an external website.

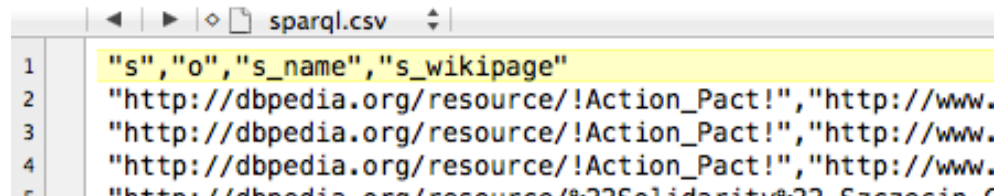
Elements: HTMLContent, FrameHeight

[HITLayout]

Reusable template to provide HITs. It is created by creating a project on the website (MTutk Web Interface).

4. Creating a Batch

- Create an input file `myhits.input`:
 - Use **tab** as separator (very sensitive)
 - First row corresponds to the variables used in the task template. In this example: `${o}`, `${s_name}` and `${s_wikipedia}`



	s	o	s_name	s_wikipedia
1				
2				
3				
4				
5				

- Each line contains the input data to generate the tasks

5. Publishing the HITs

- Run the `loadHits` command. In the example:

```
./loadHITS.sh -label PATH/MTurkDemo/cmd/myhits  
                -input PATH/MTurkDemo/cmd/myhits.input  
                -question PATH/MTurkDemo/cmd//myhits.question  
                -properties PATH/MTurkDemo/cmd/myhits.properties
```

- The following output files could be generated:
 - **Success file:** Contains identifiers to the HIT that were created. This information is used to retrieve the results of these HITs.
 - **Failure file:** Contains the rows of the input files that could not be published in the platform.

6. Retrieving and Reviewing the Results

- The results are retrieved with the `getResults` command. In the example:

```
./getResults.sh -successfile PATH/MTurkDemo/cmd/myhits.success  
                  -output PATH/MTurkDemo/cmd/myhits.results
```

Success file containing the identifiers of the HITs created with `loadHITs`

- The output file contains a column named “reject” which should be marked in case the assignment should be rejected.
- To approve/reject HITs execute the `reviewResults` command. In the example:

```
./reviewResults.sh -resultsfile PATH/MTurkDemo/cmd/myhits.results
```

MTURK API



Select a Programming Language



<http://aws.amazon.com/code/SDKs/695>



<http://aws.amazon.com/code/SDKs/922>



Ruby

<http://aws.amazon.com/code/SDKs/793>



<http://aws.amazon.com/code/SDKs/923>

Java API:

1. Pre-requisites

- Download the MTurk SDK for Java
 - <http://sourceforge.net/projects/mturksdk-java/files/latest/download>
- Include all the *.jar files from the /lib and /lib/build in your classpath
- Set up the mturk.properties file
- Include the following classes:
 - `com.amazonaws.mturk.service.axis.RequesterService`: Establishes the connection to the platform
 - `com.amazonaws.mturk.util.ClientConfig`: Connection specifications (path to the mturk.properties files)
 - `com.amazonaws.mturk.service.exception.ServiceException`: Handles the exceptions when contacting the platform
 - `com.amazonaws.mturk.requester.HIT`: Class for HIT management.

Java API:

2. Setting up the connection

- Create a RequestServiceObject
- Specify the configurations with a PropertiesClientConfig object. This class has two different constructors with the following arguments:
 - No arguments: the configuration is set up using the methods setAccessKeyId, setSecretAccessKey, setServiceURL.
 - String specifying the path to the property file mturk.properties

```
service = new RequesterService(  
    new PropertiesClientConfig("./mturk.properties"));
```

Java API:

3. Creating HITs (1)

There are different ways to create HITs using the Java API:

1. Reading certain information about the HITs from **CSV files** (properties, question, input) – same files used in the CLT example.
2. Specifying certain the information about the HITs using **Java objects**
 - This method is more suitable –specially for the input specification– when consuming data directly from other applications, for example, the results of a SPARQL query.
3. Combining 1. and 2.

Java API:

3. Creating HITs (2)

1. Reading from CSV Files

```
// Locations of files containing HIT information.
String inputFile = "./src/site_category/site_category.input";
String propertiesFile = "./src/site_category/site_category.properties";
String questionFile = "./src/site_category/site_category.question";
...

// Reading the information from each file.
HITDataInput input = new HITDataCSVReader(inputFile);
HITProperties props = new HITProperties(propertiesFile);
HITQuestion question = new HITQuestion(questionFile);

// Specification of output files.
HITDataOutput success = new HITDataCSVWriter(inputFile + ".success");
HITDataOutput failure = new HITDataCSVWriter(inputFile + ".failure");

// The method service.createHITs publishes the HITs in the platform.
HIT[] hits = service.createHITs(input, props, question, success, failure);
```

Creates bulk of HITs

Java API:

3. Creating HITs (3)

2. Creating Java Objects

```
// Specification of a question (String). It could be any form of Mturk questions:  
// QuestionForm, ExternalQuestion, HTMLQuestion or even a simple question  
// created with RequesterService.getBasicFreeTextQuestion("question here")  
String question = "<HTMLQuestion xmlns=...>...</HTMLQuestion>";
```

```
// Specification of the properties.
```

```
String title = "Information about artists.";
```

```
String description = "Help us to select the most representative picture of an artist.";
```

```
double reward = 0.04;
```

```
int assignments = 3;
```

```
// The method createHIT creates and publishes a single HIT using default values
```

```
// for the HIT properties not specified in the parameters.
```

```
// There are two createHIT methods with different arguments.
```

```
HIT hit = service.createHIT(title, description, reward, question, assignments);
```



Create an individual HIT

```
// We need to keep track of success/failure while publishing the HIT.
```

```
// Print in success file: hit.getHITId() + "\t" + hit.getHITTypeId();
```

Java API:

3. Creating HITs (4)

3. Combining 1. and 2.

```
// Specification of a question (String). It could be any form of Mturk questions:  
// QuestionForm, ExternalQuestion, HTMLQuestion or even a simple question  
// created with RequesterService.getBasicFreeTextQuestion("question here")  
String question = "<HTMLQuestion xmlns=...>...</HTMLQuestion>";  
  
// Reading the specification of the properties from a file.  
String propertiesFile = "./src/site_category/site_category.properties";  
HITProperties props = new HITProperties(propertiesFile);  
  
// The method createHIT creates and publishes a single HIT using default values  
// for the HIT properties not specified in the parameters.  
// There are two createHIT methods with different arguments.  
HIT hit = service.createHIT(  
    props.getTitle(),  
    props.getDescription(),  
    props.getRewardAmount(),  
    question,  
    props.getMaxAssignments());  
  
...
```

Java API:

4. Retrieving the Results

```
// Read from the success file the id of the HITs which were effectively published.
// Each row of the success file is as follows: HITid \t HITTypeID.
String successFile = "./src/site_category/site_category.success";
HITDataInput success = new HITDataCSVReader(successFile);

// Read each line of the success file to get the hitId.
for (int i = 1; i < success.getNumRows(); i++) {

    // Retrieve the information of submitted (completed) assignments for the HIT.
    Assignment[] a = service.getAllSubmittedAssignmentsForHIT(success.getRowValues(i)[0]);

    // Iterate over each assignment to get the answer.
    for (Assignment assignment : a) {
        // Print the results and the id of the assignment in the .results file.
        // The results of each assignment are stored in assignment.getAnswer();
        // The API offers methods to parse the XML answer.
    }
}
```

Java API:

5. Approving/Rejecting a Work

```
// Read from the output file the assignments to accept/reject.
String resultsFile = "./src/site_category/site_category.results";
HITDataInput results = new HITDataCSVReader(resultsFile);

// Read each line of the results file.
for (int i = 1; i < results.getNumRows(); i++) {
    Map<String,String> row = results.getRowAsMap(i);
    String reject = (String)row.get("reject");

    // If the column reject is not marked -> accept assignment.
    // Otherwise -> reject assignment and submit a new (available) assignment.
    if (reject.equals("")) {
        service.approveAssignment(row.get("assignmentid"), "Thank you");
    } else {
        service.rejectAssignment(row.get("assignmentid"), "Sorry");
        HIT hit = service.getHIT(row.get("hitid"));
        service.extendHIT(row.get("hitid"), hit.getNumberOfAssignmentsPending()+1, (long)3600.00);
    }
}
```


Choosing the Right Tool

Tool Comparison Table



Web Interface



Command Line Tools



API

Creating and managing your work

Start with our sample HTML templates



Create HITs visually with an HTML editor



Create and manage your HITs in batches



Support tab-delimited input files



Manage HITs created via the CLT or API



Define HITs in XML



Host HITs on your own server



Can be integrated into back-end systems



Create notifications indicating when HITs are updated



Source: https://requestersandbox.mturk.com/tour/choose_the_right_tool

MTURK SUMMARY

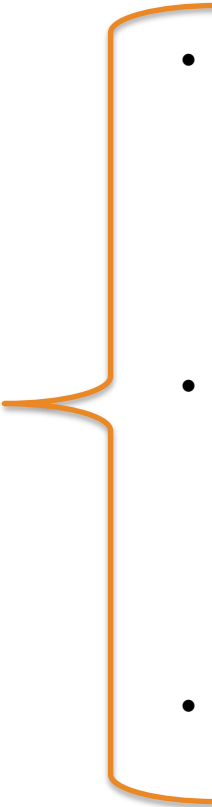
Project/HIT Creation & Design (1)

- The requester is able to create **projects** or **individual HITs**
- Build **user-friendly interfaces** (using web technologies)
- Then, the **HIT properties** must be specified:
 - **General information**: includes the title and description of the HIT, as well as keywords which are used by worker for searching HITs.
 - **HIT duration time**: time allotted to solve the HIT (before it is given to another worker).
 - **HIT life time**: how long will the HIT be available on the platform.
 - **# Assignments**: number of different persons that will perform the same HIT.
 - **Reward**: payment for correctly solving each assignment.

Project/HIT Creation & Design (2)

- Selection of **MTurk quality control** mechanisms:

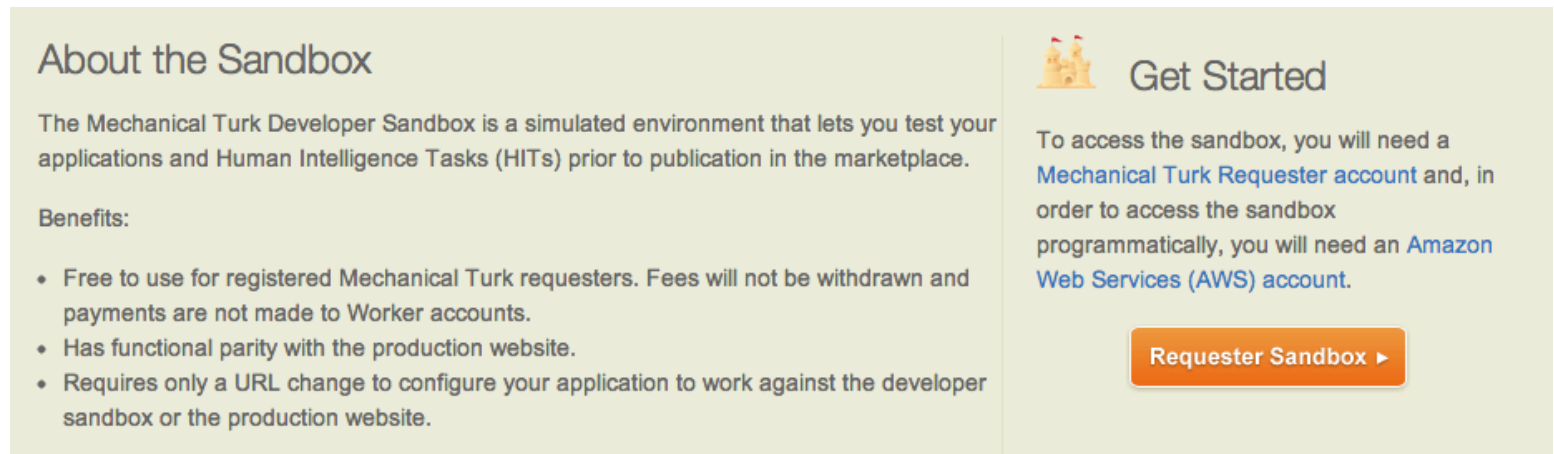
Worker
requirements

- 
- **High quality workers**
 - Masters
 - Photo moderation masters
 - Categorization masters
 - Masters expect higher rewards
 - MTurk charges 20% for masters
 - **System qualifications**
 - Location by country
 - HIT submission rate (%)
 - HIT approval/rejection rate (%)
 - (Absolute) Number of HITs approved
 - **Qualification types**
 - Simply granted or attributed via customized tests

- These filters are automatically performed by the platform

HIT Test

- **Best practice:** Always test your HITs before publishing them
 1. Perform **technical tests** (both as requester and worker) in the MTurk Sandbox environment.



The screenshot shows the 'About the Sandbox' and 'Get Started' sections of the Mechanical Turk Developer Sandbox page. The 'About the Sandbox' section describes the environment and lists benefits. The 'Get Started' section provides instructions on how to access the sandbox and includes a 'Requester Sandbox' button.

About the Sandbox

The Mechanical Turk Developer Sandbox is a simulated environment that lets you test your applications and Human Intelligence Tasks (HITs) prior to publication in the marketplace.

Benefits:

- Free to use for registered Mechanical Turk requesters. Fees will not be withdrawn and payments are not made to Worker accounts.
- Has functional parity with the production website.
- Requires only a URL change to configure your application to work against the developer sandbox or the production website.

Get Started

To access the sandbox, you will need a [Mechanical Turk Requester account](#) and, in order to access the sandbox programmatically, you will need an [Amazon Web Services \(AWS\) account](#).

[Requester Sandbox ►](#)

Source: <https://requester.mturk.com/developer/sandbox>

2. Publish a small subset of tasks in the production site to test **usability** and **responsiveness**.

Run live HITs

- **HIT publication:**
 - Make the HITs available to the workers
- **Review of the results:**
 - Monitor the submitted assignments constantly
 - Download the results
 - Accept/reject assignments, provide feedback when rejecting
 - Block spammers (optional)
- **Update HIT/Project:**
 - Extend/expire HITs or modify other HIT properties
 - Add additional assignments

IS THERE MORE THAN MTURK?



CrowdFlower Platform



- **Client:** Creates and submits jobs (MTurk = requester)
- **Contributor:** person who solves the jobs (MTurk = worker)
- **Job:** unit work (MTurk = task)
- CrowdFlower is a mediator between clients and contributors or other labor markets via different channels

Why CrowdFlower? (1)

Neat UI

Comparing content between two pages

Instructions ▲

Instructions

[Jump to questions](#)

In this task, you will help us verify whether the links to external pages contained in Wikipedia articles are correct or not. The content of these external pages should be related to the content of the Wikipedia article. In this task, you will verify whether this is the case or not. In order to solve this task, we will provide the Wikipedia article and an external website that the article links to.

Your job: Compare whether the Wikipedia article and the external website are related.
Try to **refresh the page** if the content is not displayed properly.

Example of **incorrect** data

In the following example, we are checking whether the external web pages are related to the Wikipedia article "John Two-Hawks". The following link shows a website not related to "John Two-Hawks".

Why CrowdFlower? (2)

Creation of Gold

Allows for easily creating a “gold standard”, which is further used to detect low quality workers

Job 261779 Comparing content between two pages Not Ordered

Overview Data Edit Gold Analytics Skills Reports

[Edit Gold](#)

[Judgment Stats](#)


[Gold Stats](#)

Unit #333574812 [Show job instructions](#) ☐ Copy unit to source ☐ Dig randomly [Skip](#)

The content in External page corresponds to the topic covered in the Wikipedia article?

About: !Action Pact!

Wikipedia article about: `${s_name}`



WIKIPEDIA
The Free Encyclopedia

- [Main page](#)
- [Contents](#)
- [Featured content](#)
- [Current events](#)
- [Random article](#)
- [Donate to Wikipedia](#)

Create account [Log in](#)

Article [Talk](#) [Read](#) [Edit](#)

!Action Pact!

From Wikipedia, the free encyclopedia

"Action Pact" redirects here. For the album by Sloan, see [Action Pact \(album\)](#).

!Action Pact! were a [punk rock](#) band, formed in 1981 as the Bad Samaritans by guitarist Wild Planet, bassist Dr. Phibes, and drummer

!Action Pact!	
Also known as	The Bad Samaritans
Origin	Stanwell, London, England
Genres	Punk rock

Why CrowdFlower? (3)

Report generation and analytics

Job 230234 Categorization of fashion images

Finished ▼

Overview Data Edit Gold Contributors Analytics Skills Reports

Summary

Contributors

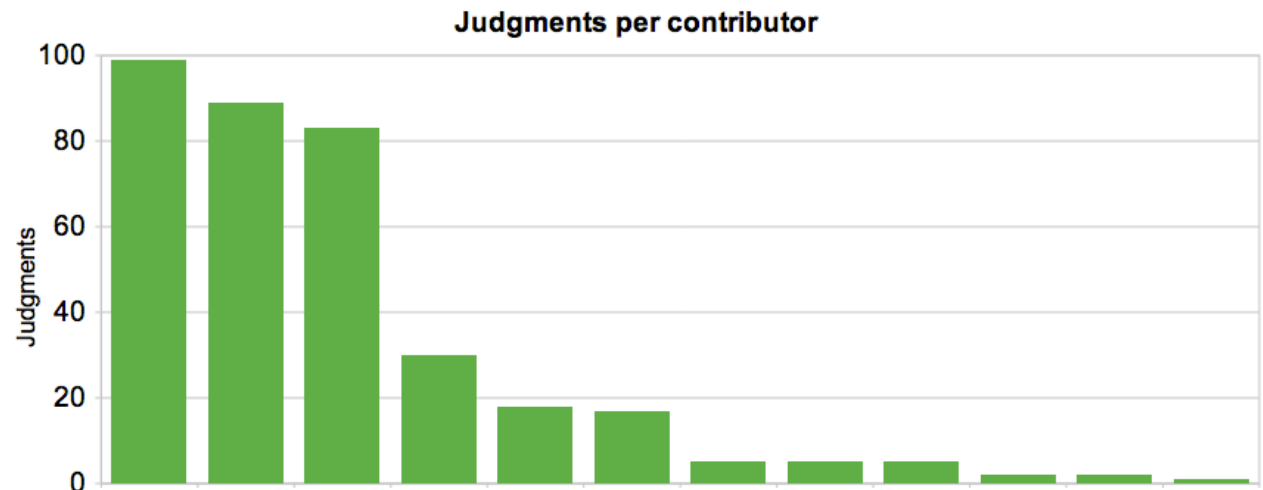
Quality

Gold

Distributions

Times

Each bar represents a contributor and the number of judgments they have submitted for this job. Contributors who have a low trust score and have submitted a significantly larger amount of judgments than other contributors are likely scammers. If your gold is working correctly, their work will be rejected. (Only the top 100 contributors are displayed in this graph.)



Each bar represents a contributor

Why NOT CrowdFlower?

- At the beginning, clients must wait until their **projects are approved** by the CrowdFlower staff before publishing them
 - Wait time: From a couple of hours up to (5)* days
- Jobs must be specified in a **non-standard language**:
 - CML: CrowdFlower Markup Language
- There are certain **configurations** that cannot be executed in the platform

References

- AMT. Getting Started Guide. API Version 2012-03-25
<http://s3.amazonaws.com/awsdocs/MechTurk/latest/amt-gsg.pdf>
- The Mechanical Turk Blog
<http://mechanicalturk.typepad.com/>
- MTurk Java API
<http://people.csail.mit.edu/glittle/MTurkJavaAPI/>
- CrowdFlower Platform
<http://people.csail.mit.edu/glittle/MTurkJavaAPI/>