Coding Guide for Detailed Tool Use Coding 2023

Coiba National Park – Cracking Capuchins Project
Zoë Goldsborough & Meredith Carson

This is a guide detailing how to code tool use sequences of white-faced capuchins using coding-software *BORIS*. This guide serves the purpose of providing step-by-step explanations for first time users, and as a reference guide for more advanced coders. It includes the full ethogram and subject list with key bindings as a reference.

The very first step is to **install** and **launch** BORIS. It can be installed from here (https://www.boris.unito.it). For the BORIS manual see here (https://boris.readthedocs.io/en/latest). BORIS runs best on Windows, but is supported on Mac as well (albeit not the latest version).

Table of contents

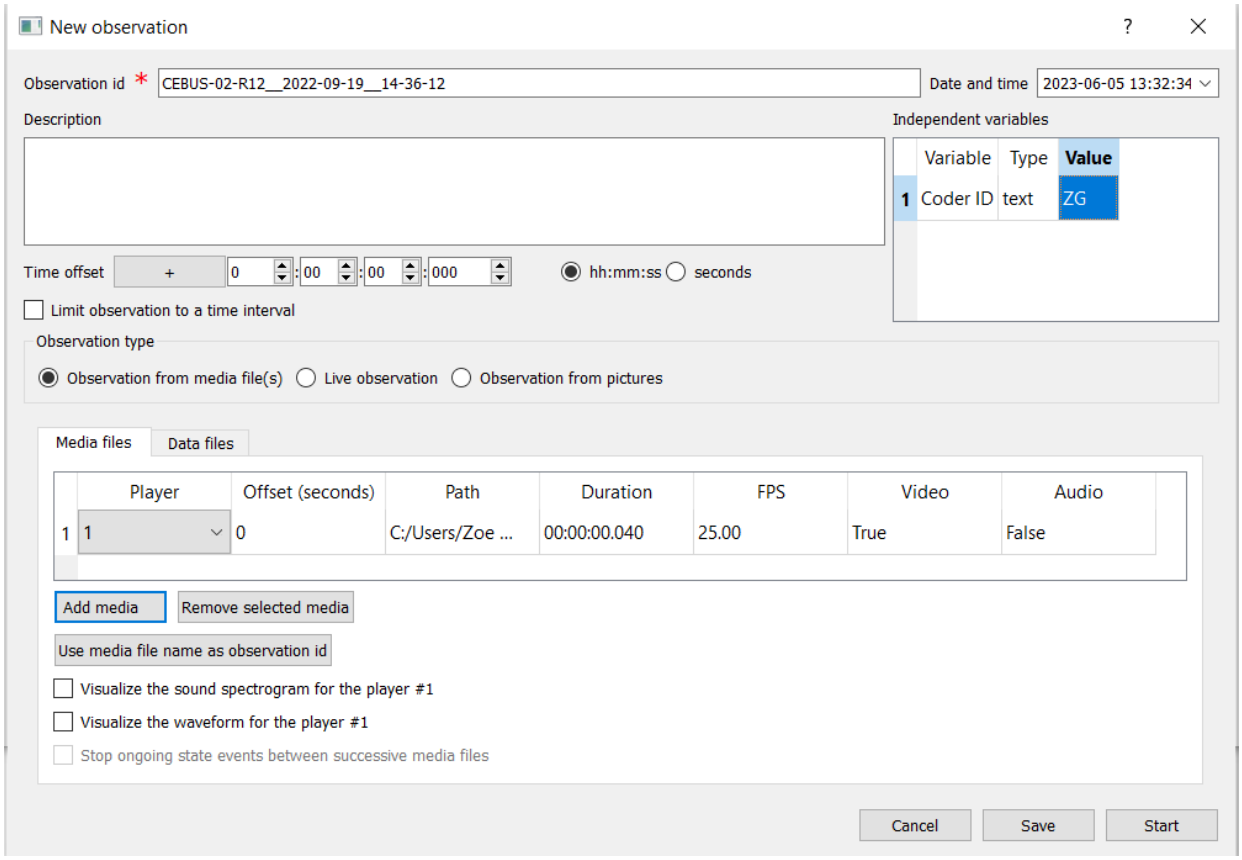
Loading the project

Once BORIS has been launched, you need to open the right project, which contains all of the necessary information and coding protocols. Our project is called “Cracking Capuchins – Detailed Tool Use Coding”. You will receive a Project file for you to work in, which will be named “Cracking Capuchins_[YOUR INITIALS].boris”.

Changes to the project can be made by pressing “Edit Project” under the File dropdown menu, but this should not be necessary. If changes are necessary, this will be communicated to you.

Creating a new observation (video)

1. Press CTRL+N (or Cmd + N on Apple) or “New Observation” from the Observations dropdown menu
2. Press “add media” to select the video you want to code
3. **ObservationID:** if you’re on the latest version of BORIS, then press the button “Use media file name as observation ID”, then take off the .MP4 extension. E.g. CEBUS-02-R11__2022-01-30__07-09-32
   Note: if you are not yet on the latest version of BORIS, you have to copy/paste the filename into the observation ID box yourself.
4. **Independent variables:** specify who the coder is by entering your initials in the coder box (highlighted blue in the screenshot below)
5. Press “Start” to begin coding

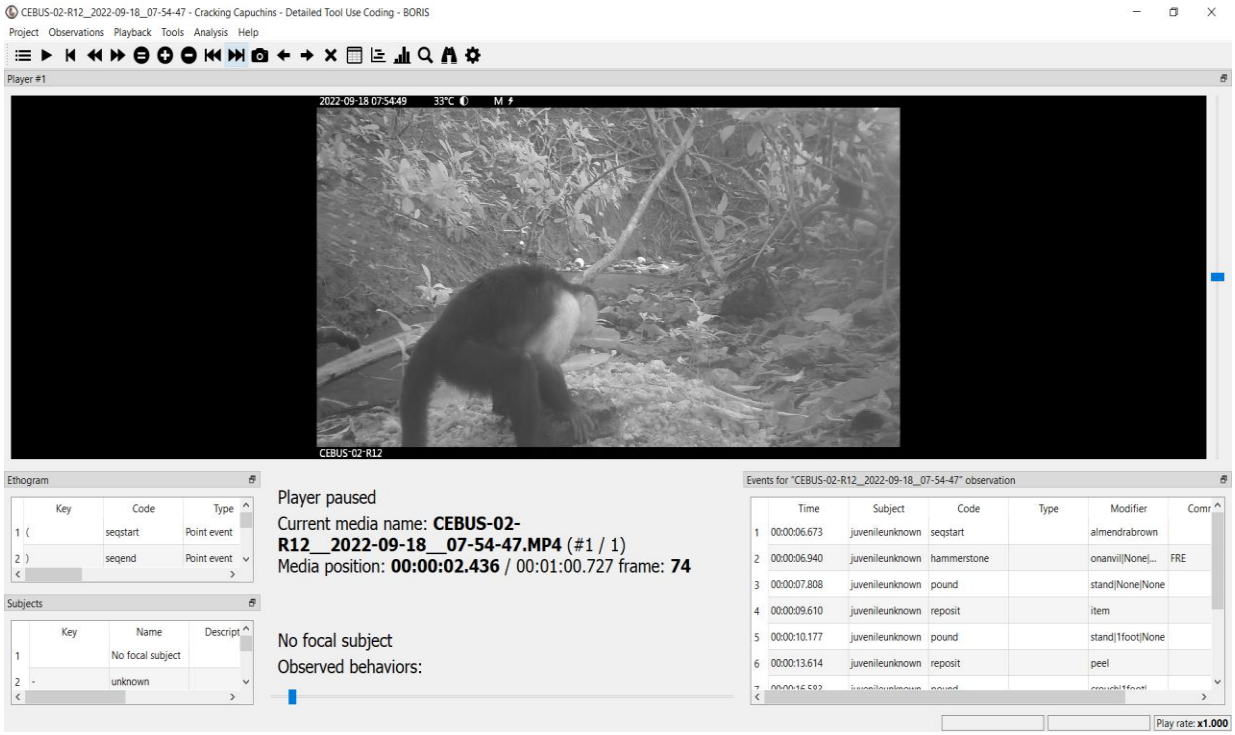


Before coding the behaviors, I recommend watching the video once in its entirety to get an idea for who is processing what and what takes place, then you can go back to the beginning and start coding. For coding purposes, playing the video at slower speed or frame-by-frame can help. In the latest version of BORIS, the video plays at normal speed (and pauses when you press “spacebar”) but switches to frame-by-frame automatically once you press the left or right arrow key.

Coding tool use sequences

A **video** is one **observation,** but we are interested in **tool use sequences**: one video can contain multiple sequences (and a tool use sequence can span two videos). A sequence runs from when a capuchin first grabs an item to open with tools and places it on the anvil until it finishes processing it.

If it all worked out okay, your screen looks something like this:



Some things to note:

- At the top you see the video that you are coding, by pressing the windows icon on the right top you can get a new window with the video (the same goes for the other places where this icon occurs)
- On the left bottom is your **ethogram**, which has all the possible behaviors (and descriptive information) with their shortkeys
- In the middle bottom you see your **current progress**, so where in the video you are, what mediafile this is, and who the current focal subject is.
- On the right bottom we have the **events scored,** an event is when you record an instance of a behavior, so here you can see (and edit) events you have

already recorded. E.g. in this example I have coded a pound by an unknown juvenile at 00:00:07.808, with as modifier that it was a standing pound.
- You can change the size of the different areas by going to the edge of them with your mouse and dragging your mouse to make them larger or smaller.
- At the very top you see buttons that allow you to control a variety of things in the program. Let's get into the most useful ones.

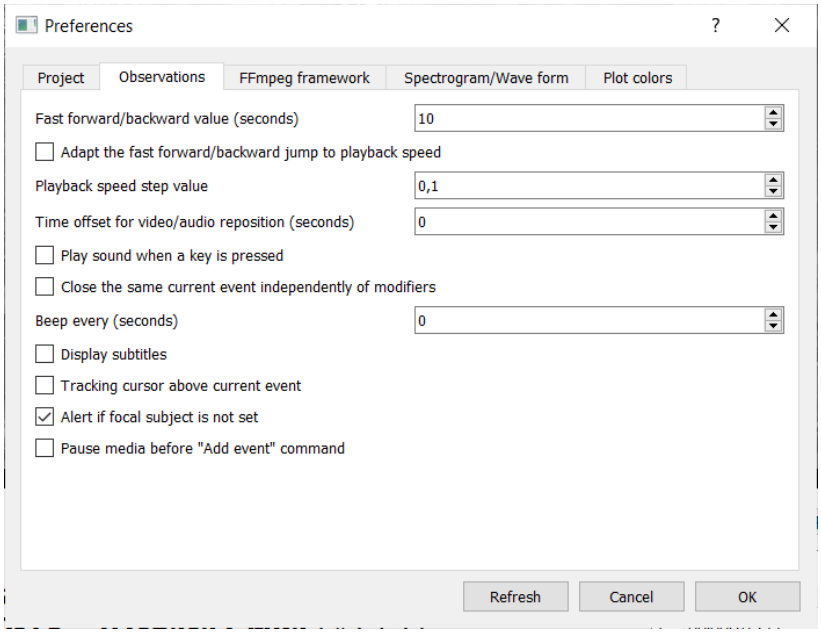
Controls:

- **Space Bar:** play or pause the media
- **Page Up key:** switch to the next media (not as relevant for us)
- **Page Down key:** switch to the previous media (again not relevant)
- **Up arrow key:** jump forward (10 seconds) in the current media
- **Down arrow key:** jump backward (10 seconds) in the current media
- **Home key (or + symbol in top bar):** Increase the playback speed
- **End key (or – symbol in top bar):** Decrease the playback speed
- **Backspace:** Set the playback speed to 1x
- **Left arrow key:** go to the previous frame
- **Right arrow key:** go to the next frame

You can get into **frame-by-frame** mode, which might be necessary to see the capuchin's behavior in detail, by pressing the **film reel icon. Note:** in the latest version of BORIS, you automatically go into frame-by-frame once you press left or right arrow key.

Another relevant button is the **picture camera**, which allows you to take a **snapshot** of the current video or frame.

Lastly, select the **gears icon** to change any preferences, such as the jump forward speed. I recommend ticking the box "Alert if focal subject is not set".



Behavior coding step-by-step:

One thing you need to understand is the concept of **modifiers:**

If you press the key of a behavior (e.g., a pound, with "q") a pop-up window appears with additional information you can enter about this behavior. This is optional, but often it is essential information so please always look at it carefully to see if anything applies. You can either select modifiers by clicking on the one you want, or pressing its linked key (e.g. "f" for an one-footed pound). You can press several keys and the window will only disappear once you press "OK" or the Enter key.

Notes:

- **Comments:** You can add comments to keep track of rare events (e.g., the anvil fracturing or if they use something else than the experimental anvil). You add a comment by right-clicking an event you already recorded in the list and then pressing "edit". Then you can type in the comment box
- **Editing and deleting events:** you can always fix or delete events that you coded wrong by pressing them in the events list in the right-bottom area. You can also edit multiple events at the same time (e.g., change the subject of all of them, or add a comment to all) by selecting them all and then right-clicking and pressing edit selected events.
- **Sorting:** You can sort the ethogram or subjects list by pressing the column name, e.g., by key by pressing "key".

When you identify the start of a sequence, go through the following steps:

1. Enter the **focal subject** of the sequence (so the tool using individual) by pressing the appropriate key. This can either be a specific individual ID if you recognize them, or a number for unidentifiable individuals which specifies their age and/or sex. The way BORIS is set up it remembers your subject so once you have specified it you can enter behavior without re-entering the subject. If you have not specified a subject you get a popup (if you have ticked the box in preferences!).
2. As the very **first** behavior in the sequence, press **"("** to signify the start of the sequence and enter the additional information that pops up as modifiers. This has to be the first thing coded for each sequence!
3. After letting it run for a moment, or progressing a few frames with the right arrow key, press **"h"** to add information on the hammerstone and again fill out all the modifiers that pop up (Note: you can either fill out the hammerstone's end location now if you already watched the whole sequence, or code it at the end of the sequence. I usually fill out "e_onanvil" as this is the most common end location, and then go back to change it if it wasn't the case). Avoid coding hammerstone at the exact same timestamp as sequence start, but always later, because otherwise it might end up in front of it and fall outside of the sequence in data processing.
4. Now code all the relevant behaviors as they happen. For the list of possible behaviors with their modifiers see the next few pages. Remember to always make sure you have the correct focal subject specified.

5. If you did not yet code the ending location of the hammerstone, press "**h**" and enter only that modifier, or alternatively, go back and change the ending location when you coded "h" the first time. When you reach the end of the sequence, press "**)**" to code the sequence end as the **very last thing in the sequence**. Again, enter the additional information that pops up as modifiers. In case the video ends before the capuchin is done processing the item, you can also enter this. See the detailed ethogram for more information.
6. When you are done, **save** the observation (CTRL + S, or in the dropdown menu the button "save project"). You can also continue an observation you started at a later moment by pressing "start observation" from the Observation dropdown menu and picking the observation from the list.

Very important: always code the sequence start as the very FIRST thing in the sequence and sequence end as the very LAST thing!! This is very crucial for cleaning the data later. So make sure no other behaviors share the same timestamp with sequence start or end.

Exporting data

Since BORIS has a tendency to crash sometimes, and backing up a project file can be difficult (some software considers the *.boris* extension to be a virus), I recommend exporting your observations after each coding session and backing up the resulting csv file. Exporting goes as follows:

1. Press "Export Events" and then "Aggregated Events" in the Observations dropdown menu
2. Select one or multiple (or all) of the observations you wish to export
3. Leave the subjects and behavior checkmarks as is and leave "full observations" checked.
4. When you get the popup "Group events from selected observations in one file?" click "Yes"
5. You get a popup of file explorer where you can specify the filetype. Enter a filename that includes your initials and save it as **csv.**
6. Upload the csv with the raw data to https://drive.google.com/drive/folders/1JBZpnERZADLNo3A6stKxx1kSKVTKRFPR?usp=sharing

Reference Guide

Subjects

Coding identifiable individuals

Important-- Only ID an individual capuchin if you feel positive. If you have a tentative clue of an individual, you can put a note in the comments. Use the syntax below--indicating you think, but are unsure, it is ABE.

ID?=ABE

Note: if your video contains a clear face-shot of the tool-using individual, but you can't recognize them, please make a comment that says "face-shot" so we can later revisit this sequence and identify them retroactively!
Yellow highlighted ones are individuals added only during the 2022 coding (so could be older versions of 2017-2019 coded individuals). Blue highlight means the key is not intuitive (e.g., not as easy as "a" for Abraham).

Pictures here:
https://docs.google.com/presentation/d/1NIHzr3YFCGftdW2K8fei7kYjRuhGfpbzXJkKzF6eFug/edit#slide=id.p10

Name	Code	Key	Sex	Age (~2022)
Abraham	ABE	a	Male	Adult
Snaggletooth McGee	SMG	s	Male	Adult
Cystopher	CYS	c	Male	Adult
Mr. Email	MRE	m	Male	Adult
Tom	TOM	t	Male	Adult
Beatrice	BEA	b	Female	Adult
Olga	OLG	o	Female	Adult
Leona	LEO	l	Female	Adult
Dottie	DOT	d	Female	Adult
Sadie	SAD	**e**	Female	Adult
Rick	RIC	**k**	Male	(Sub)adult
Ink	INK	i	Male	Adult
Joker	JOK	j	Male	Subadult
Spot	SPT	**p**	Male	(Sub)adult
Yoda	YOD	y	Male	Subadult
Larry	LAR	**r**	Male	(Sub)adult
Mick	MIC	**?**	Male	Subadult
Balthasar	BAL	**.**	Male	Juvenile
Frida	FRI	f	Female	Subadult
Peaky	PEA	**^**	-	Juvenile
Zim	ZIM	**~**	-	Juvenile
Terry	TER	**>**	Male	Old juvenile/young subadult
Joe	JOE	**x**	-	Juvenile

Coding unidentifiable individuals

Age-Sex	Key
Adult male	1
Adult female	2
Subadult male	3
Subadult female	4
Juvenile male	5
Juvenile female	6
Unknown male	7
Unknown female	8
Juvenile unknown	9
Unknown	-

The most important information for this project is the **age** of the individual. Therefore, whenever possible try to at least determine if they are adults or juveniles! **Important** is that we **only see males use tools** at this site, so very likely all tool users will be male!

Some additional information on aging/sexing capuchins, with examples (courtesy of Brendan Barrett).

Adult males are easy to identify because of their size. They are bulkier and often have wider heads than other capuchins. They will typically be 6-7 years or older. They typically are much larger, have more protruding/snouty faces, and can be balder on the forehead, which increases with age sometimes. This sometimes reveals splotches or dark skin there the cap meets the forehead that is useful for identifying individuals. They also often have scars particularly if they are older. Also, if you can see testicles or perhaps clear evidence of a penis, that is a good hint they are a male.

3 adult males with a juvenile in foreground.

Adult females, when they are older than 10, they often get very fluffy ridges of hair above their eyebrows/forehead. If they are lactating, you can sometimes see longer nipples or swollen mammary glands. They also often have less snouty faces. Young adult females (5 years or older) typically do not have the eyebrow ridge, but sex can be figured out by smaller size and shape. They may have denser hair on the forehead compared to males. Clear evidence of nursing infants is a useful hint for sex. A Female capuchin's clitoris can be very large and may be mistaken for a penis, particularly in subadults and juveniles.

Adult female

Subadults: Males and Females at around 2.5-5 years old, although a subadult male and small adult female can often be the same size. It is best to rely on genitals to identify these individuals, particularly if you are new to this. Subadult males are less filled out and bulked up than adult males, but considerably larger than juveniles (and often have more snouty faces/larger foreheads already)

Juveniles: Individuals 0-3 years old. Are noticeably smaller than adults, may also be dorsal infants. These are much harder to sex, so we do not often sex them unless we see the genitals clearly.

Juvenile male processing an almendra. Note white tuft of hair next to right ear.

Left to Right: Subadult male, adult male, juvenile (male i think)

Sequence Independent Variables

Name	Key	Modifier Names	Modifier Options	Definition
seqstart	(			Start of tool use sequence, defined as the moment when the capuchin first places the item on the anvil. If the item is already on the anvil at the start of video, the start of the sequence equals the start of the video. **Note:** if capuchins are processing items on another anvil than the experimental anvil (e.g. a branch) make a comment with seq_start saying "wooden anvil" **Note:** if a sequence is continued from a previous video DO NOT code seqstart again! This is only for the very beginning of a sequence.
		itemtype	1. Almendra green (or yellow) 2. Almendra brown 3. Almendra unknown 4. Halloween crab 5. Hermit crab 6. Coconut 7. Fruit 8. Other (add comment) 9. Unknown 10. Almendra red (**r)**	Code which item is being processed in the tool use sequence as a modifier.
seqend	)			The end of the tool use sequence, defined as the moment the capuchin starts consuming the item (if opened) or otherwise when they let go off the hammerstone with no further strikes on the item.

		success	1. Opened 2. Relocated 3. Abandoned 4. Continued	Add modifier to specify how the sequence ended. If the capuchin is eating the item, code it as "opened". The other options are the capuchin taking the item elsewhere, or abandoning it. "Continued" is an indicator that the sequence is not yet finished when the video ends. Will help stitch together sequences that span multiple videos. In case the video ends and you don't know what happened to the item (i.e., it's not recorded), keep seqend at "none" and add a comment saying it was unknown/missed. **Note:** if they open the item but do not eat it, then code "abandoned" with a comment saying "opened but not eaten".
		scrounging	1. scrounging (s) 2. no scrounging (n)	Code this behavior if any scrounging (other individuals eating parts of the item opened by the tool user) occurred during or after the sequence. **Important:** only code no scrounging if other individuals were present but there was no scrounging. If no other capuchins are visible, then leave it blank at "none".
		displacement	1. No displacement (x) 2. Anvil displacement (a) 3. Hammer displacement (h) 4. Full displacement of both hammer and anvil (f)	At the end of the sequence, did any displacement occur? If so, was the capuchin displaced just from the anvil or hammer or both? If displacement occurred, make a comment with the ID or age/sex of the displacing and displaced individuals. E.g.: "LAR displaces JOE" or "Subadult male displaces juvenile"

				Important: only code no displacement if other individuals were present but there was no displacement. If no other capuchins are visible, then leave it blank at "none". **Note:** only code displacement once. For example, if a juvenile is using tools and gets displaced, then code displacement for the juvenile's tool use sequence, but not for the sequence then started by the individual that did the displacing.
		social attention	1. no social attention (z) 2. social attention (t)	At the end of the sequence, did any other individuals pay attention to processing by the tool-user? Attention means coming close to the anvil and peering at the tool-user while they are processing (does not have to include scrounging). **Important:** if there was no opportunity for social attention (i.e., no other individuals around) then leave it blank at "none" and **do not** code no social attention.
hammer stone	h			To collect relevant information about the hammerstones
		hammerloc_s	1. In hand (i) 2. On anvil (o) 3. Off anvil within reach (r) 4. Off anvil walk (w) 5. Carry in (c)	The location of the hammerstone first used for the tool use sequence **relative to the experimental anvil!** Code the location where the hammerstone is ***at the beginning of the sequence***. "In hand" means the capuchin is already holding it, while "on anvil" means it is lying on the anvil and off anvil can be either within reach

				(within 1 body length of the anvil) or at a walking distance (>1 body length). Additionally, the hammerstone can be carried in from out of view.
		hammerloc_e	1. on anvil (a) 2. off anvil within reach (h) 3. off anvil out of reach (f) 4. carry out (t)	Code the hammerstone location again at the **end of the sequence**. Now the location can only be on the anvil, off anvil within reach (within 1 body length of the anvil), off anvil at a further distance or carry out if they take it with them.
		hammerID	Enter the hammerstone ID as a comment, or if it's not marked/ unidentifiable: **7.** Unmarked 8. Unknown	Comment the ID of the hammerstone first used for the sequence, which can be one of the marked hammerstones or in the case of an unmarked hammerstone code "unmarked" but describe it in the comments.

Hammerstone IDs

Pictures here
https://docs.google.com/presentation/d/1l_2cutk85T0_oC-6oMh2W73EFKwZUF85HzQeFSiM80E/edit#slide=id.p

Location	Name	Deploy	Code	Description
EXP-ANV-01	Dwayne	R11 & R12	DWA (later DWA_A and DWA_B)	Two orange spray paint stripes. Large, gray, rounded all over. **Fractures** into Dwayne A and B
	Dinner Plate	R11	DPL	Two orange spray paint stripes. Round and flat like a plate.
	Big Chunk	R11	BCH	Rough edges and single orange stripe
	Little Chunk	R11	LCH	Rough edges, angular and blocky with single orange stripe. Smaller than Big Chunk
	Mjolnir	R11	MJO	Dark, near-black material. **Not marked,** found on EXP-ANV-01 in July. Dark, near black material.
CEBUS-02	Pebbles	R11, R12	PEB	Three orange stripes. Rounded edges with boomerang shape
	Fred	R11, R12	FRE	Very large, three orange stripes. Blocky, rectangular
	Wilma	R11, R12	WIL	Medium size (larger than PEB, smaller than FRE). Rounded, smoothed and two orange stripes (one thick, one thin)
	BamBam	R11, R12	BAM	Very small (smaller than PEB, FRE and WIL), three orange stripes, thin, flat and hat-shaped outline.

Behaviors

Behavior	Key	Modifier Name	Modifier Options	Definition
pound	q			The capuchin hits the item with the hammerstone successfully, code all modifiers and code pound at (approximately) the moment the hammerstone hits the item.
		poundtype	1. Crouching 2. Standing 3. Jumping	We differentiate between three types of pounds. It is a crouching pound if the legs of the capuchins are at around a 90-degree angle. If the legs are extended beyond 90 degrees (and often the body is elongated too), it is a standing pound. Once one of the feet (can be both) leaves the ground it is a jumping pound.
		position	1. 1foot (f) 2. 1hand (h) 3. tail support (t) *Multiple selection possible*	Modifiers describing the position of the capuchin. The default is 2 footed (both feet on the ground) and 2 handed (both hands on the hammerstone). Only code tail support if you can see the capuchin clearly using their tail (i.e., gripping

				something with it or putting weight on it).
		hammer	1. Overhead (o)	Modifier to indicate that the hammer was held above the capuchin's head during the hit, so whether any part of the stone is higher than their head at the peak of the pound.
reposition	**z**			When the capuchin repositions the item on the anvil (so anytime they touch/move the item in the sequence) or clearly changes their grip on the hammerstone. Code every time it occurs, but not every movement (e.g. if they grab the item and move it 3 times rapidly, code reposition once). **Note:** don't code the first time placing/adjusting the item or gripping the hammerstone. We are only interested in *re*positioning.
		objecttype	1. hammer (h) 2. item (i) 3. peel (p)	Whether it is the hammer or item that is repositioned. The third option is that they "peel" the item by holding it in their hand and

				manipulating it with hands or teeth (often tearing pieces off). **Note:** don't code peel when they do it at the end and eat the item straight after without more pounds in between.
misstrike	w			When capuchins do not hit the item as intended, or something else goes wrong. You can often use audio cues to hear if they strike the item or the anvil. **Note:** if capuchin does hit the item successfully but also has a misstrike (e.g. item flies off) then code both the pound and the misstrike.
		mistaketype	1. item flies off (i) 2. drop hammerstone (h) 3. hammer break (b) 4. anvil break (d) 5. other (o)	Specify what occurred, either no hit at all (the default), the item flies off the anvil, the hammerstone falls out of their grip, it breaks or the anvil breaks. If not in this list then select other and add a comment to the behavior. **Important:** also add a comment if you see a capuchin injure themselves with the hammer.

hammerswitch	**g**			When a capuchin grabs another hammerstone during the sequence.
		hammerloc	1. On anvil (o) 2. Off anvil within reach (r) 3. Off anvil walk (w) 4. Carry in (c)	The location of the hammerstone they switch to.
		hammerID	Enter the hammerstone ID as a comment, or if it's not marked/ unidentifiable: **7.** Unmarked 8. Unknown	Enter the ID of the hammerstone first used for the sequence as a comment, which can be one of the marked hammerstones or in the case of an unmarked hammerstone code "unmarked" but describe it.
anvilswitch	**v**			When individuals switch from one anvil to another that are both within view of the camera, within a sequence. So are processing something and then continue to process **the same item** at a different location.
		anvilmaterial	1. wood (w) 2. stone (s)	The material of the anvil that they **switched to**