Purpose

The Reveal component simplifies building gamified experiences using Wyng, both for clients relying on no-code solutions, as clients building enhanced and customized solutions incorporating custom code.

This guide provides an overview of how the Reveal component works, the configuration options and assets, and the SDK functions available for front-end developers.

1. Functional overview

The Reveal component initially displays some configurable interactive content, e.g., a gaming interaction like a Spin-to-Win or Scratch Off. After a user interacts with it, the component reveals one of a set of outcomes, either randomly or determined by the user action. The component includes basic participation limit logic.

2. Quick start examples

To create a Spin-to-Win that shows random outcomes:

- Create a new experience in Wyng.
- Drag and add a Reveal component from the component toolbar.
- For Pre-Reveal, select the Spin-to-Win interaction.
- Upload the assets for Spin-to-Win.
- Set how many different outcomes you want.
- Set the outcome selection mode to "Random"
- Assign a probability to each outcome.
- Map outcomes of the Spin-to-Win wheel to each sector.
- Drag and add a Text component into the slots for each outcome.
- Set up the content for each outcome in the Text components.
- Preview and publish your experience.

To create a Spin-to-Win that shows codes from a linked Reward component:

- Create a new experience in Wyng.
- Drag and add a Reveal component from the component toolbar.

- For Pre-Reveal, select the Spin-to-Win interaction.
- Upload the assets for Spin-to-Win.
- Set the outcome selection mode to "Linked Reward". This creates a single outcome slot.
- Drag and add a Reward component to the outcome slot.
- Upload a code list to the Reward component.
- Rewards requires a form submit to allocate a code:
    - Drag and add a Form component from the component toolbar.
    - Link the Reward component to the Form.
- Set up the content for each scenario in the Rewards component, for code issued, no code issued, and reminder cases (if a winner comes back again).
- Map code labels and no code issued case to sectors of the Spin-to-Win wheel.
- Preview and publish your experience.

3. Pre-Reveal state

The component is initially in ***Pre-Reveal*** state, where some interactive content is displayed. You may select from some productized gamified interactions developed by Wyng, or add your own custom code. In the initial release of Reveal, a ***Spin-to-Win*** productized interaction is available; further details are provided later in this guide.

If you choose to develop a custom experience, you may add HTML, JavaScript, and CSS code within the Reveal component, that will be displayed within the component container when it is in Pre-Reveal state. After the desired user interaction, you can trigger a transition to one of the outcomes using an SDK function. SDK functions are provided to transition to a random outcome, or to any specific outcome. For example, here is code for a simple custom experience that selects a random outcome when the user clicks a button, using an SDK function:

```
<button onclick="window.wyng['_WYNG_ID_'].revealSelectOutcome();">
    Click Me!
</button>
```

To limit or gate participation, for example, to limit a user to one participation per session, or to gate interaction with a form submission, the Pre-Reveal component may be in ***Locked*** or ***Unlocked*** state. If locked, any productized interaction like Spin-to-Win will be disabled; if unlocked, the user will be free to participate.

For custom experiences, you can use the SDK to add an event listener callback for transitions between locked and unlocked based on the participation limit feature. Alternatively, you can disable the participation limit feature, and implement custom logic and state to limit participation using the revealUnlock() and revealLock() SDK functions.

4. Outcome selection

After the desired interaction, the component transitions from the Pre-Reveal state to display an outcome, e.g. a reward code, a click-to-redeem link, or form that someone can complete and submit to claim a reward.

There are three modes for selecting outcomes:

1) *Random selection.* A probability is set for each outcome in configuration, and outcome selection after participation is random, weighted by the defined probabilities.
2) *Custom selection*. For custom Pre-Reveal experiences, a specific outcome can be selected by a SDK function call.
3) *Selected by linked Rewards component*. For programs allocating unique coupon codes to each participant, a Reward component can be linked. For this case, there is only 1 outcome slot allowed, and a Reward component should be added to that slot. The Pre-Reveal is locked until the Reward component issues a code or a "no code available" end state. Once the Reward end state is reached, the Pre-Reveal is unlocked, and the user can participate.

After an outcome is selected, and the Pre-Reveal interaction or animation is complete, the outcome content is displayed. The outcome content can either replace the interactive content or be displayed just below the interactive content. Visually, there are CSS transitions for both cases, which may be overridden with custom CSS.

5. Outcome content

You can configure any number of distinct outcomes; at least one outcome is required, as a minimum. The content for each outcome consists of a slot, into which you can add any component from the Experience editor: for example, outcomes can display HTML components, Text components, or Forms. Different outcome slots may have varying component types.

For linked Rewards, there is only 1 outcome slot. The Rewards component needs to be added to that slot. In this case, the content displayed when codes are issued or if no codes are available can be set up directly in the Rewards component.

Front-end developers have more flexibility over how to display outcomes to a user, by adding an event listener on outcome selection. The event handler can take custom actions like navigating the user to a different sub-page of the experience for each outcome, or hiding and revealing other components.

Special case: If no component is added to an outcome slot, the Pre-Reveal content remains visible, even after the outcome is selected. This may be useful for custom experiences where an

event listener is handling outcome display; for example, a Spin-to-Win could remain visible on the page, while the outcome is displayed by revealing a component just below it.

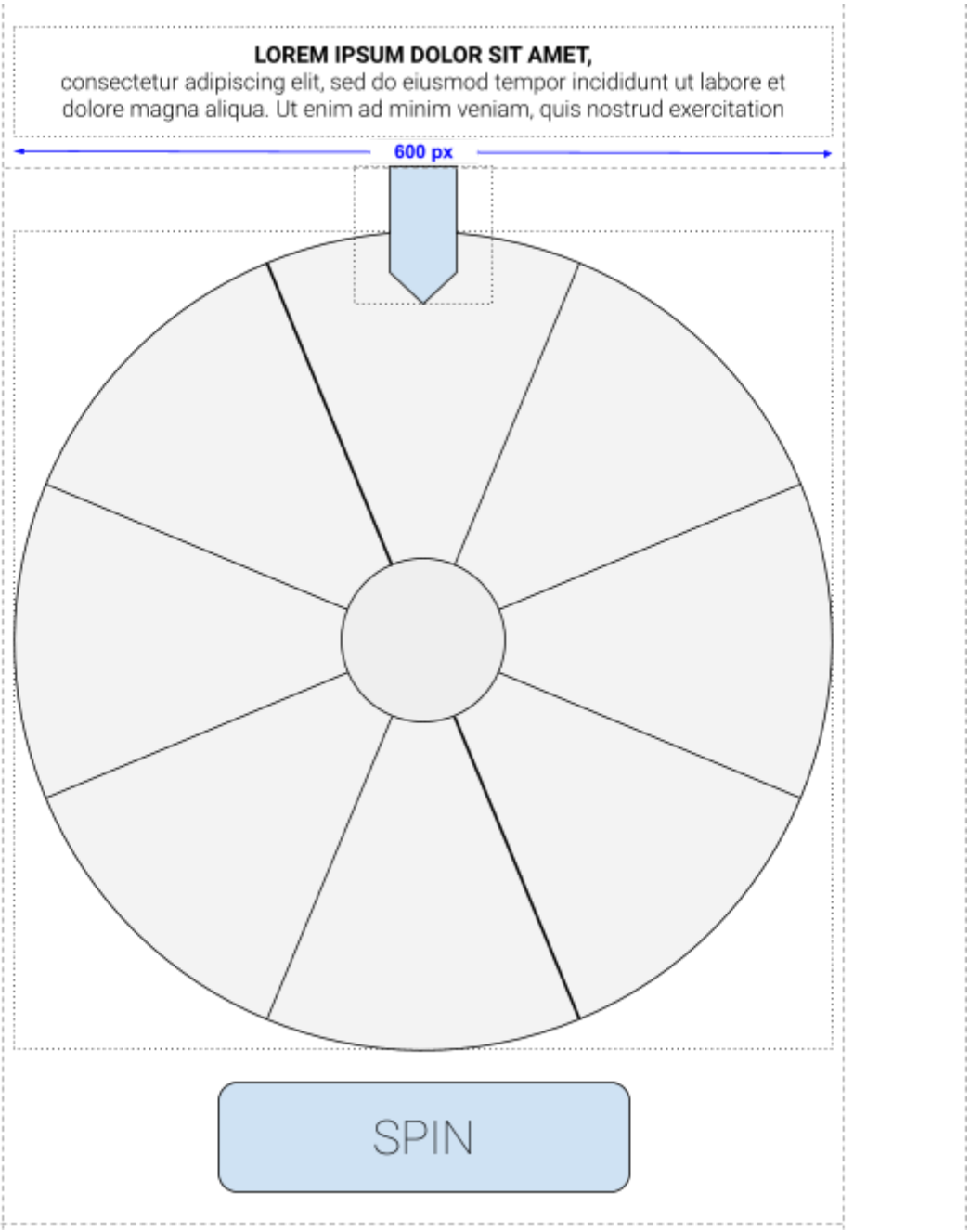
6. Participation limit

The Reveal component provides a "once-per-session" participation limit, by default. Once the user has interacted and an outcome is selected and revealed, if the participant refreshes the browser or leaves the page and returns during the same web session, the outcome is "sticky"; that is, the component remembers and immediately transitions to the same outcome previously selected. This application state is kept in browser localStorage.

You can choose to disable the default participation limit; if disabled, when the participant refreshes the page, they can participate again. In this mode, you can implement your own custom logic using SDK functions.

7. Wireframe

Pre-Reveal State: Spin-to-Win



8. Configuration settings

Setting	Values	Description
Participation limit	Once-per-session, Disabled	Default is once-per-session: once an outcome has been selected and revealed, if the participant refreshes or returns, the same outcome is immediately revealed. If disabled, the user can participate again on refresh, and the SDK can be used to implement custom behavior using the revealLock() and revealUnlock() functions.
Outcome selection mode	Random, Custom, Linked Reward	Use the "Custom" option to select specific outcomes using the revealSelectOutcome() SDK function. Select "Linked Reward" to associate the Reveal component with a Reward component. The Reward must be added to an outcome slot.
Pre-Reveal interaction type	Spin-to-Win, Scratch-Off, Custom	Select from available productized interactions or implement your own using HTML, JavaScript and CSS.
[Spin-to-Win] Wheel image	Upload image asset	Wheel image with any number of evenly spaced sectors. The first sector should start at the exact top of the wheel (the zero degree position). 600px X 600px, PNG format, transparent outside circumference of wheel. The wheel will initially be positioned so the middle of the first sector is at the top center: rotated counterclockwise by half a sector (360 degrees / number-of-sectors) * 0.5.
[Spin-to-Win] Pointer image	Upload image asset	Pointer image that will be overlaid on top of the wheel at the zero degree position. The pointer points down, with the tip at the bottom-center of the asset. 100x x 100px, PNG format, transparent outside edges of pointer.
[Spin-to-Win] Number of sectors	Choose a number from 2 to 20	How many evenly sized sectors are in the wheel image.

[Spin-to-Win] Show button	Yes, No	Participants click the button to spin the wheel. The wheel keeps spinning as long as the participant is pressing the button, and then slows to a stop. For custom experiences, the button can be omitted, to use SDK functions to spin and stop the wheel.
[Spin-to-Win] Button label	Enter text, e.g. "Spin"	
[Spin-to-Win] Sector mapping	[Not Linked Reward] List of outcomes names; map each to a sector, using a select list. [Linked Reward] Add one or more code labels, and map each to a sector, using a select list. Plus: Map the "no code available" case to a sector.	Each outcome name is mapped to a sector of the wheel. For the Linked Reward case, each code label and no code available is mapped to a sector of the wheel. Also map which sector to use if an unmapped code label is returned by the reward. **Follow-up**: If Code service can return a list of distinct code labels, for the Linked Reward case, the list of code labels is generated automatically.
[Scratch-Off] Overlay image	Upload image asset	Overlay to scratch off with mouse or finger, to reveal outcome. 600px x 600px recommended, PNG format. The image is sized *object-fit:contain* on the Reveal component. Once more than 50% of the image is scratched off, the overlay is removed.
[Scratch-Off] Background color	Color picker	Any space around the overlay image within the Reveal component container is filled with this solid color.
[Scratch-Off] Min-height	Height in px or %	Applies to the Reveal component. This should be set so the component height is large enough to fit all of the outcome content, so the component does not resize when an outcome is selected.
List of outcomes	Enter a list of 1 or more outcome names.	This defines how many outcomes, and names them. Outcome names must be unique. They are not displayed to a participating user, but they are visible in metrics, and used in SDK events. If the Outcome selection mode is "Linked Reward", this setting is hidden (or visible but hard-coded to 1

		outcome, named "Reward").
Outcome display mode	Replace pre-reveal content OR Below pre-reveal content	Determine the position of the outcome content. The content can either replace the pre-reveal or appear just below it. When displayed, there is a CSS transition, which can be overridden with custom CSS.
[Per outcome] [Random mode] Probability	Enter a numerical weight. Floating point numbers allowed.	If "Outcome selection mode" is random, a probability weight needs to be assigned to each outcome. The probability of any given outcome is weight-of-outcome/(sum of all weights).
[Per outcome] Slot	Drag and drop a component into slot	Slots will be created in the Experience editor for each outcome, and Text, HTML, Form or other components can be dragged into each slot. Each slot shows the name assigned to the outcome as a placeholder. A slot may be left empty.

9. SDK functions for custom experiences

Event listener callback when Reveal is unlocked By default, when the session participation limit is enabled, Reveal is unlocked immediately after the session state is checked	reveal_unlocked	Payload: • eventType • componentId
Event listener callback when Reveal is locked By default, when the session participation limit is enabled, Reveal is locked after an outcome has been displayed	reveal_locked	Payload: • eventType • componentId
Event listener callback when an outcome is selected In many cases, the outcome may be selected before the	reveal_outcome_selected	Payload: • eventType • componentId • outcomeName

user interacts with the pre-reveal content. For example, in Scratch-Off, the outcome must be selected before the user starts to scratch off the top layer. In Spin-to-Win, the outcome must be selected before or while the wheel is spinning, to determine at what sector the wheel stops.		If linked reward: • reward.code • reward.label
Event listener callback when an outcome is displayed For Scratch-Off, this occurs after >50% of the overlay is transparent. For Spin-to-Win, this occurs after the wheel lands on a sector.	reveal_outcome_displayed	Payload: • eventType • componentId • outcomeName If linked reward: • reward.code • reward.label
Event listener callback when an outcome is redisplayed, if a user refreshes after being shown an outcome.	reveal_outcome_redisplay	Payload: • eventType • componentId • outcomeName
Unlock Pre-reveal content Overrides default lock/unlock state for custom gating of when a user can participate.	revealUnlock()	Parameters: • componentId
Lock Pre-reveal content Overrides default lock/unlock state for custom gating of when a user can participate.	revealLock()	Parameters: • componentId
Select an outcome. If the Outcome selection mode is "random", outcomeName is not required, and ignored if present. If the Outcome selection	revealSelectOutcome()	Parameters: • componentId • outcomeName

mode is "custom", outcomeName is required.		
Start spinning the wheel, reaching set speed, and continue spinning. Use this for experiences with separate buttons to start and stop, or behavior where the wheel keeps spinning as long as a button is being pressed.	revealSpinWheel()	Parameters: ● componentId ● speed (optional)
If it is not spinning at the set speed already, start spinning the wheel and reach set speed. Sustain speed for duration. Then slow to a stop and display the selected outcome.	revealSpinWheelAndStop()	Parameters: ● componentId ● speed (optional) ● duration (optional)
Remove the scratch-off overlay	revealScratchComplete()	Parameters: ● componentId
Reset outcome state and put component back in pre-reveal state. This can be used for a "Try Again" feature, for example.	revealResetOutcome()	Parameters: ● componentId
Get the outcome state. This can be used to access the outcome for display in custom experiences.	revealGetOutcome()	Parameters: ● componentId Returns: ● componentId ● outcomeSelected If outcomeSelected: ● outcomeName If linked reward: ● reward.code ● reward.label

10. Metrics

A new metrics card will help clients understand user interaction with Reveal: How many visitors participated (clicked “spin” or started scratch off), how many visitors saw an outcome, and the breakdown of different outcomes delivered to visitors.

New reporting events generated:

- When the user begins interacting with the pre-reveal animation. The trigger is specific to the animation type – for Spin-to-Win, this is triggered when the user clicks the “Spin” button. The event includes the animation type (“Spin to Win”, “Scratch Off”). The “Participation started” event collection is used. This event is only generated for the pre-defined animation types implemented in the component. For custom HTML pre-reveal, developers need to trigger a reporting event using an SDK function. We will add that function as a follow up.
- When an outcome is made visible to the user. The event includes the outcome type, and the outcome mode (“reward”, “random”, “custom”). In the case of “reward”, the “code label” is used as the outcome type. In the “random” and “custom” cases, the outcome name is used as the outcome type. These events are sent to the “Participation completed” event collection.
- If there are any hyperlinks added to custom HTML content or outcome content, clicks and click-through events are generated; this is consistent with how they are currently tracked in HTML or Text components.

New metrics card shows:

- A count of participation starts
- A count of participation completes
- Percent of participation completes (completes/starts)
- Pie chart of outcome types