addons presta

Cookie Consent Mode v2

USER MANUAL · Version 1.0.0 · PrestaShop 1.7.8 – 9.x

1. What this module does

A fast cookie banner with Google Consent Mode v2, without any external consent service. The module shows your visitors a cookie banner with three buttons of equal weight — **Accept all**, **Reject** and **Settings** — and a dialog where they choose by category. It remembers the choice in a cookie of your own shop and tells Google (Consent Mode v2), Meta Pixel, TikTok Pixel and Microsoft Advertising (UET) what the visitor allowed. Everything runs from your shop: no subscription, no external service, no request to another server.

Works with **PrestaShop 1.7.8, 8.x and 9.x** (PHP 7.2.5 or newer), with the Classic and Hummingbird themes and with multistore.

2. Installation

1. Download the module from **My account** → **Order history** at addonspresta.com.
2. In your Back Office, open **Modules** → **Module Manager** and click **Upload a module**.
3. Drop the `ap_cookieconsent_v1.0.0.zip` file and wait for the confirmation.
4. Click **Configure** on the module card.

If you have shell access, you can also unpack the zip into `modules/` and run `php bin/console prestashop:module install ap_cookieconsent` from the shop folder.

Right after the installation the banner is shown as a bar at the bottom of the page, in every language of the shop, with default texts you can change. All four consent signals are on, the choice is remembered for 6 months, and the module has been moved to the first position of the `displayHeader` hook in every shop, so that the consent default reaches the tags before they start.

After the installation, check these points:

1. **Is the module first?** At the bottom of the configuration page, the **Script order** panel must say **First** for every shop. If it does not, click **Move to the first position**. You can see the same in **Design** → **Positions**, hook `displayHeader` (tick **Display non-positionable hooks** when the hook is not listed): Cookie Consent Mode v2 must be the first module.
2. **Choose your policy page** in **Policy page** and read the default texts in the **Texts** panel. They are a starting point; adapt them to the cookies your shop really uses.

3. **Add the module's cookie to your cookie policy.** The cookie is called `ap_cookieconsent`. It stores the shop number, the consent version, the day of the choice (not the time) and three yes/no answers (preferences, analytics, marketing) — nothing that identifies the visitor. It is kept for 6 or 12 months from the day of the choice, as set in **Remember the choice for**.
4. **Clear the caches**: in **Advanced Parameters** → **Performance** click **Clear cache**, and empty the cache of your page cache module or proxy (Varnish, Cloudflare and similar) if you use one.
5. **Test the signals with Google Tag Assistant** (tagassistant.google.com): connect your shop, open the **Consent** tab and check that the "On-page default" is denied before you answer the banner and that an "On-page update" follows your choice. In the browser console, `dataLayer` starts with the `consent` `default` entry, and `apCookieConsent.get()` shows the stored choice.

3. How it works

The banner. A visitor who has not chosen yet sees the banner: your title and text, the link to your policy page and the buttons **Accept all**, **Reject** and **Settings**, all three with the same style, so that refusing is as easy as accepting. As a bar at the bottom of the page the banner does not block the shop; as a dialog in the centre it covers the page until the visitor chooses. **Settings** opens the preferences dialog, titled **Cookie settings**, with four categories:

- **Necessary** — always on, marked **Always active**, cannot be switched off;
- **Preferences** — cookies that remember settings and choices;
- **Analytics** — traffic measurement;
- **Marketing** — advertising and measuring campaigns.

Before the first choice no category except Necessary is ticked. The visitor saves with **Save choices**, **Accept all** or **Reject**. The banner and the dialog can be used with the keyboard alone: they come first in the keyboard order, the dialog keeps the focus inside while it is open, Escape closes the preferences dialog without a choice and the focus goes back to where it was. The banner has no animation and takes its fonts, colours and buttons from your theme.

The consent signals. A small script printed in the page header, before the tags, sends the "denied" default on every page and, once the visitor has chosen, the update — on every following page too, straight after the default.

- **Google** (Google Analytics, Google Ads, Google Tag Manager): Consent Mode v2 default with `ad_storage`, `ad_user_data`, `ad_personalization`, `analytics_storage`, `functionality_storage` and `personalization_storage` denied and `wait_for_update` of 500 ms. After the choice, Marketing grants the three advertising types, Analytics grants `analytics_storage` and Preferences grants `functionality_storage` and `personalization_storage`.
- **Meta Pixel**: the consent is revoked until the visitor accepts Marketing, then granted.
- **TikTok Pixel**: the consent is held until the visitor chooses, then granted or revoked according to Marketing.
- **Microsoft Advertising (UET)**: `ad_storage` denied by default, updated according to Marketing.

A tool that is not installed in your shop simply ignores its signal. Tags inside Google Tag Manager follow the choice when they are set up to use Google's consent types (built-in consent checks or additional consent settings of the tag).

Why the order matters. The default has to reach Google before any Google tag starts, so the module must be the first module in the `displayHeader` hook, where modules such as Google Analytics (ps_googleanalytics) print their tags. The installation puts it there and the **Script order** panel keeps an eye on it. Being first in the hook is not the same as being first in the page header, though: whatever is pasted directly into the theme's `head.tpl` file above the hook runs earlier. A Google tag or Tag Manager snippet pasted there starts before the consent default; let a module print it in `displayHeader`, or move it below the hook.

Meta Pixel. PrestaShop's own module PrestaShop Social (ps_facebook) is covered: it asks before every page view whether the pixel may run, and this module answers. A pixel added with Meta's standard code, directly or through a tag manager, is held as long as its code comes after this module's script. A pixel loaded in another way, or injected after the page has loaded while a script of the theme has declared its own `fbq` variable above the hook, can send the first page view of a visitor who has not chosen yet. The fix is always the same: Meta's standard code, below this module.

The cookie. The choice is stored in the cookie `ap_cookieconsent`, written by the browser on your shop's own domain. It contains the shop number, the consent version, the day of the choice and three yes/no answers — every visitor who chose the same categories on the same day has exactly the same value. The choice is valid for 6 or 12 months counted from that day, so up to one day less than the setting. When the consent version of the shop changes, or the period is over, the visitor is asked again. Safari keeps cookies written by scripts for 7 days only; the module writes the cookie again on every visit, so visitors who come back within a week keep their choice for the whole period, and those who come back less often are asked again.

Page caches. The page code is the same for every visitor, so page cache modules and proxies can store it. The choice exists only in the visitor's browser. After you change texts, switches or the consent version, clear the page cache, otherwise visitors keep getting the old version.

What the module does not do.

- It does not block, delay or remove scripts. It sends consent signals to the four tools above; any other script of your shop keeps running as before. A developer can tie other scripts to the visitor's choice (see "FAQ and troubleshooting").
- It does not scan your shop for cookies and does not keep a log of consents.
- It is not a certified consent management platform (CMP) of Google and does not use the IAB TCF framework. Google asks for a certified CMP from sites that show its ads (AdSense, Ad Manager, AdMob) to visitors in the EEA.
- It does not make your shop compliant by itself and is not legal advice. The texts of the banner and of your policy are yours; check with your adviser what your shop needs.

Without JavaScript nothing is shown and nothing is granted. If the module's script cannot be printed (for example because a theme override of its template is broken), the shop sends the denied defaults to all four tools, shows no banner and writes a message to **Advanced Parameters → Logs**.

The module stores only its settings. It does not change PrestaShop tables, does not connect to addonspresta.com or any other server and sends no data out of your shop; the consent signals go only to the tags that are already on your pages.

4. Features and settings

Open **Modules → Module Manager → Cookie Consent Mode v2 → Configure**. The **Help** box at the top links this manual and the module page at addonspresta.com in the language of your Back Office, gives our support address and shows the installed version. Below it is one form with three panels — **Banner**, **Consent signals** and **Texts**; every **Save** button saves all three — and the **Script order** panel.

Layout

- **Bar at the bottom of the page** (default) — the visitor can use the shop while the bar is shown.
- **Dialog in the centre of the page** — the dialog covers the page until the visitor chooses; it cannot be closed without a choice, and **Reject** is always one click away.

On the policy page itself the banner is always a bar, so that the policy can be read.

Remember the choice for

6 months (default) or **12 months**. After this time the visitor is asked again.

Consent version

A number from 1 to 999999, 1 after the installation. When you change it, every visitor is asked again at the next visit. Raise it after you add a new tool to your shop or change your cookie policy.

Policy page

The CMS page the banner links to (**Design → Pages**). The list shows the active CMS pages of the shops of the current context that you manage; **No link** removes the link. Describe the cookies of your shop on that page, including the module's own cookie `ap_cookieconsent` (see "Installation").

Link "Cookie settings" in the footer

Shows a **Cookie settings** link in the footer, a single line above the copyright in Classic and Hummingbird, so that visitors can open the preferences dialog and change their choice at any time. On by default. To place the same link elsewhere, see "FAQ and troubleshooting".

Google Consent Mode v2

Sends the default and the update to Google Analytics, Google Ads and Google Tag Manager. On by default.

Meta Pixel

Holds Meta Pixel until the visitor accepts marketing cookies. On by default.

TikTok Pixel

Holds TikTok Pixel until the visitor accepts marketing cookies. On by default.

Microsoft Advertising (UET)

Sends the consent for advertising cookies to the Microsoft UET tag. On by default.

The four switches are on after the installation because a tool that is not in your shop ignores the signal, and a tag added later is covered at once. Switch one off only when another solution sends that tool its consent.

Banner title

The title of the banner, for each language. By default "We use cookies".

Banner text

The text of the banner, up to 1000 characters. The default text says that the shop uses necessary cookies and, with the visitor's consent, cookies for preferences, traffic measurement and marketing, and that the choice can be changed at any time in the cookie settings.

Text of the policy link

The words of the link to your policy page, shown when a page is chosen in **Policy page**. By default "Privacy and cookie policy".

Necessary cookies

The description shown under **Necessary** in the preferences dialog. By default it names the cart, login, checkout and remembering the choice.

Preference cookies

The description shown under **Preferences**.

Analytics cookies

The description shown under **Analytics**.

Marketing cookies

The description shown under **Marketing**.

Rules for all texts. Every text exists in each language of the shop and is required in the default language; a text left empty in another language uses the default language. At the installation every language gets the module's text — English, Polish, French, German, Spanish or Italian, English for other languages. Titles and the link text can have up to 100 characters, the category descriptions up to 500. The texts are plain text: the characters `<` and `>` are refused, line breaks become spaces, and no-break spaces you type (French punctuation, for example) are kept.

Button labels and category names — **Accept all**, **Reject**, **Settings**, **Save choices**, **Cookie settings**, **Necessary**, **Preferences**, **Analytics**, **Marketing**, **Always active** and **Close** — are translations of the

module. Change them in **International → Translations**: choose **Installed modules translations**, the module Cookie Consent Mode v2 and your language.

Script order

The panel shows, for every active shop of the current context that you manage, the place of the module in the `displayHeader` hook:

- **First** — everything is in order;
- **After:** followed by the modules that run before it — their tags may start before the consent default;
- **Not attached to the hook. Attach the module in Design > Positions.** — the module prints nothing in that shop.

Move to the first position (shown while the module is not first somewhere) puts the module first in those shops. It only moves: in a shop where the module was detached from the hook or is disabled, it does not attach it again — a warning names those shops instead. Attach it there in **Design → Positions** with **Hook a module** (**Transplant a module** in PrestaShop 1.7 and 8), then move it first.

5. Multistore

Settings are saved for the shop or shop group selected in the Back Office header. Select **All shops** to set the same values everywhere, or a single shop to give it its own texts, policy page, layout and switches. In the All shops or group context only an employee who manages every shop of that context can save.

Each shop also keeps its own visitor choice: the cookie carries the shop number, so a shop in a subfolder of another shop's address (`/shop2/`) asks its visitors separately. The **Script order** panel and **Move to the first position** work on the shops of the current context that you manage. One licence covers every shop of the installation.

6. Uninstall

Module Manager → Cookie Consent Mode v2 → Uninstall deletes the settings and the texts. The module has no table to delete. The `ap_cookieconsent` cookie stays in your visitors' browsers until it expires.

Disable keeps the settings, but while the module is disabled there is no banner and no consent signal at all: your tags then run without any consent default. Disable it only for a short check, or when another solution takes over.

7. FAQ and troubleshooting

The banner does not show. Most often the choice is already stored in your browser: the banner comes back only when the choice expires or the consent version changes. Open the preferences with

the **Cookie settings** link in the footer, or use a private window. If the banner is missing for new visitors too, clear the cache in **Advanced Parameters → Performance** and your page cache, and check that the module is enabled and attached to `displayHeader` (**Script order** panel).

Tag Assistant shows a Google tag before the consent default. The tag is printed above the module. Look at the **Script order** panel first; if it says **First**, the tag is pasted into your theme's `head.tpl` file above the hook. Let a module print it in `displayHeader` (ps_googleanalytics does), or move it below the hook.

Meta Pixel Helper shows a page view before I answered the banner. The pixel is loaded in a way the module cannot hold. Use PrestaShop Social (ps_facebook) or Meta's standard code, placed below this module: a module in `displayHeader` after it, or a tag manager.

Visitors using Safari are asked again after about a week. Safari keeps cookies written by scripts for 7 days. The module renews the cookie at every visit, so only visitors who come back less than once a week are asked again.

I changed a text, a switch or the consent version, and the shop still shows the old one. A page cache still serves the old page: clear the cache of your cache module or proxy and **Clear cache** in **Advanced Parameters → Performance**.

After an update of the module the old behaviour stays. With **Never recompile template files** selected in **Advanced Parameters → Performance**, PrestaShop keeps using the old compiled template of the consent script. Click **Clear cache** after every update of the module.

The log says "Cookie Consent Mode v2: the consent script could not be printed". A theme override of the module's `head.tpl` is broken. The shop then denies everything and shows no banner until you remove the file from `themes/<theme>/modules/ap_cookieconsent/`. Overriding `head.tpl` is not supported; the link template `link.tpl` is the one meant for themes.

Move to the first position shows a warning that names a shop. In that shop the module is not attached to `displayHeader`, or it is disabled. Attach it in **Design → Positions** with **Hook a module**, then click **Move to the first position** again.

How do I ask every visitor again? Raise the **Consent version**, save and clear your page cache.

Does the module block my other scripts, such as a chat or a heatmap tool? No. It sends consent signals to Google, Meta, TikTok and Microsoft only. Your developer can start another script only after consent with the module's JavaScript interface: `apCookieConsent.get()` returns the current choice, and the event `ap-cookieconsent:changed` fires every time the visitor saves a choice. For example, for an analytics tool (the code must come after the module's script, for example in the footer):

```
<script>
  function startMyTool(consent) {
    if (consent.analytics && !window.myToolStarted) {
      window.myToolStarted = true;
      loadMyTool();
    }
  }
  document.addEventListener('ap-cookieconsent:changed', function (event) {
    startMyTool(event.detail);
  });
  if (window.apCookieConsent) {
    startMyTool(window.apCookieConsent.get());
  }
</script>
```

`get()` returns `decided`, `necessary`, `preferences`, `analytics` and `marketing` as true or false. A script that has already started cannot be stopped by a later refusal; the tool has to handle that itself.

Can I put the "Cookie settings" link somewhere else? Yes. `{widget name="ap_cookieconsent"}` in a theme template prints the same link, and so does any other display hook you attach the module to in **Design → Positions**. Any element with the attribute `data-ap-cookieconsent-open` opens the preferences dialog on click, and `apCookieConsent.open()` does it from a script. The `README.md` file in the module zip, section "Customising", shows how to change the link and the look of the banner from your theme.

In multistore my second shop asks visitors again. That is intended: every shop has its own settings and its own choice in the cookie.

Is the module enough to comply with the GDPR and the ePrivacy rules? The module is a tool: it asks, remembers the choice and passes it on to the four tools. Which cookies your shop uses, what your texts and policy say and whether other scripts need consent is for you to check, with an adviser if needed.

8. Updates and support

Updates are free for life: download new versions from **My account → Order history** at addonspresta.com and upload them like a new module. Your settings are kept. After an update, clear the cache in **Advanced Parameters → Performance**.

Support is included for 12 months from your order: write to **mail@addonspresta.com** or use the contact form at addonspresta.com (Monday to Friday, 10:00–16:00 CET; we usually answer within 2 business days). Please include your PrestaShop version, the module version and the steps that lead to the problem.