Автозаполнение ответов из URL

Автозаполнение ответов интервью из URL-параметров (userCode)

> **Статус:** реализовано· **Модуль:** web-интервью (CAWI) · **Дата:** 2026-07-16
> **Аудитория:** методисты/администраторы опросов, интеграторы (формируют ссылки), разработчики.

Коротко

При открытии web-интервью по ссылке с параметром вида `?sex=1` приложение **само выбирает** нужный вариант ответа: находит вопрос, у которого код `userCode == "sex"`, и отмечает в нём вариант, у которого `userCode == "1"`.

Это избавляет респондента от повторного ответа на вопросы, ответы на которые уже известны заранее (пол, возраст, регион и т. п. — например, переданы панелью или из CRM).

```
https://<адрес-приложения>/?poll_id=<uuid>&user_name=<имя>&sex=1
                                                    ▲
                                        код вопроса = sex, код варианта = 1 → выбран
автоматически
```

---

1. Контракт

Правило ровно одно и симметричное:

Часть URL	Чему соответствует
Имя параметра (`sex`)	`userCode` **вопроса**
Значение параметра (`1`)	`userCode` **варианта ответа**

Сопоставление идёт **только по** `userCode`. Ни `answerCode`, ни порядковый номер, ни текст варианта в матчинге не участвуют.

---

2. Настройка опроса (методист / администратор)

Чтобы вопрос участвовал в автозаполнении, `userCode` нужно **включить и заполнить** в двух местах:

1. **На самом вопросе** — задать код (например `sex`) и включить флаг `userCode.enabled`.
2. **На каждом варианте ответа**, который должен быть выбираемым по ссылке, — задать код (например `1` для «МУЖ», `2` для «ЖЕН») и включить `userCode.enabled`.

Пример (эталонный тестовый опрос «AUTOTEST-12»):

Объект	Текст	userCode
Вопрос	«Пол»	`sex`
Вариант	МУЖ	`1`
Вариант	ЖЕН	`2`

Ссылка `...&sex=1` выберет «МУЖ», `...&sex=2` — «ЖЕН».

> **Важно:** вариант без включённого `userCode` по ссылке выбрать нельзя — он просто не будет найден. Отката на другие коды нет by design.

---

3. Формирование ссылки (интегратор)

Базовый формат web-ссылки на интервью:

```
https://<адрес-приложения>/?poll_id=<uuid-опроса>&user_name=<имя-респондента>&<код1>=<значение>&<код2>=<значение>
```

Параметры автозаполнения добавляются к обычной ссылке как дополнительные пары `код=значение`.

Несколько вопросов сразу

Каждый вопрос — отдельный параметр:

```
...&sex=1&age=3&region=77
```

Мультивыбор (вопросы с несколькими ответами)

Несколько вариантов одного вопроса — через **запятую в одном параметре**:

```
...&hobby=1,3,5
```

Использовать повторяющийся ключ (`hobby=1&hobby=3`) **нельзя** — при разборе URL остаётся только последнее значение, предыдущие теряются.

⚠ Кодирование символов — критично

Значения параметров автозаполнения берутся из URL **как есть, без URL-декодирования**. Отсюда два правила:

- **Запятую-разделитель не кодировать.** `hobby=1,3` — правильно. `hobby=1%2C3` — приложение получит буквально строку `1%2C3`, не разобьёт её на `1` и `3`, и ни один вариант не совпадёт.

- **Не рассчитывать на раскодирование `%20` и подобного.** Пробелы и спецсимволы в кодах лучше не использовать вовсе. Пробелы по краям значения приложение обрежет само (`sex= 1` → `1`), но `%20` пробелом не станет.

---

4. Правила сопоставления (важные нюансы поведения)

Ситуация	Поведение
Значение совпало с `userCode` варианта	Вариант выбирается автоматически
Строгое сравнение строк	`01` ≠ `1`, `1.0` ≠ `1`. Код должен совпасть точно (пробелы по краям обрезаются)
Мультивыбор `q=1,3`	Выбираются оба варианта, **в порядке значений из URL**
В вопросе настроен **autoSelect** (условный автовыбор)	autoSelect **главнее** — автозаполнение по ссылке для такого вопроса не применяется
На вопрос **уже дан ответ** (респондент ответил / есть черновик)	Автозаполнение **не перезаписывает** ответ
Чекбокс с лимитом (максимум N ответов)	Если по ссылке пришло больше вариантов, чем разрешено, берутся первые N (в порядке URL), остальные отбрасываются
Чекбокс с «эксклюзивным» вариантом (напр. «Затрудняюсь ответить»)	Если среди пришедших есть эксклюзивный вариант — выбирается только он
Вариант **скрыт** условием видимости	Не выбирается (скрытые варианты в автозаполнении не участвуют)
Параметр есть, но **подходящего варианта нет** (`sex=99`)	Ничего не выбирается, приложение продолжает работу, в лог пишется предупреждение
Параметра для вопроса **нет в ссылке**	Вопрос остаётся пустым, респондент отвечает сам

Ключевой принцип: **автозаполнение либо помогает, либо молча уступает**. Оно никогда не ломает интервью, не перетирает уже данный ответ и не спорит с настроенным в опросе автовыбором.

---

5. Что НЕ поддерживается

Осознанно вне текущего объёма (планируется отдельно):

- **Диапазоны возраста** (например «19 лет» → группа «18–24»). Сейчас нужно точное совпадение кода.
- **Открытый ввод** — числовые и текстовые поля по ссылке не заполняются.
- **Табличные и шкальные вопросы.**

Для этих типов вопросов параметры в URL игнорируются без ошибок.

---

6. Диагностика

Фича не имеет видимой респонденту индикации — вся диагностика идёт в **технический лог интервью** (`saveLog`). Что искать при разборе:

Запись в логе	Что значит
`нет варианта под значение параметра <код>`	Параметр в ссылке есть, но вариант с таким `userCode` не найден (опечатка в коде варианта, вариант скрыт или `userCode` выключен)
`неоднозначное совпадение`	Несколько вариантов имеют одинаковый `userCode` (ошибка настройки опроса)
`userCode prefill пропущен (D6): ... есть autoSelect`	Автозаполнение не сработало, потому что в вопросе включён autoSelect (это ожидаемо)
`вариант(ов) сверх maxAnswers ... отброшено`	Для чекбокса по ссылке пришло больше вариантов, чем разрешает лимит

Первый шаг при жалобе «ссылка не подставляет ответ»: сверить код в URL с `userCode` варианта в настройках опроса (регистр, ведущие нули, пробелы) и убедиться, что `userCode.enabled` включён и на вопросе, и на варианте.

---

7. Для разработчиков

Точка входа данных: параметры URL разбираются в `src/main.vue` (`forWebApp()`), складываются в профиль (`SET_WEB_INFO`) и доступны как `getMap`. Значения не декодируются, повторный ключ перетирается — см. раздел 3.

Ядро фичи:

- `src/assets/utils/userCodePrefill.js` — чистый резолвер `resolveUserCodePrefill(question, map)` и store-guard `isQuestionAnswered(answeredSlides, slideId)`. Без побочных эффектов, покрыт юнит-тестами.

- `src/assets/mixins/radioMixin.js` и `src/assets/mixins/checkBoxMixin.js` — метод `applyUserCodePrefill(...)`, встроенный в существующий механизм автовыбора (`checkAutoSelect` / `change`). Выбранные варианты проходят штатным путём, поэтому квоты, ветвление и отправка работают без изменений.

Тесты:

```
npm run test:usercode-prefill   # юнит-тесты резолвера (node), + фикстура реального опроса
npm run test:lifecycle          # lifecycle-тесты интеграции в mixin'ы (jsdom)
```

Оба набора включены в `npm run build` и `npm run build-web`.

Проектная документация: дизайн-спека и решения D1–D7 — `docs/superpowers/specs/2026-06-10-url-param-autofill-by-usercode-design.md`.

---

8. Проверенные сценарии (QA)

Живой прогон на опросе AUTOTEST-12 (`53fb0813`), вопрос «Пол»:

Сценарий	Результат
`?sex=1` → выбран «МУЖ»	✅ Подтверждено (store + DOM)
`?sex=2` → выбран «ЖЕН»	✅ Подтверждено
`?sex=99` (несуществующий код)	✅ Ничего не выбрано, интервью работает, предупреждение в логе
Приоритет черновика (D1)	Покрыто lifecycle-тестами; в web-флоу каждая загрузка стартует новое интервью, поэтому вживую неприменимо
Чекбокс / autoSelect (D6/D7)	Покрыто lifecycle-тестами; в эталонном опросе таких вопросов нет — для live-проверки нужен опрос с чекбокс-вопросом и вопросом с autoSelect

---

Revision #1
Created 20 November 2026 13:27:34 by Admin
Updated 20 November 2026 13:27:34 by Admin