

Forum:[Hack Forums](#)

Thread: [Quick Send Popularity](#)

Charlie Sheen - October 2, 2025, 11:38 am

This script gives you the ability to send a popularity from a post, instead of having to go to the members profile and send one that way. Right next to the members popularity stat, you'll see a thumbs up/thumbs down. Click it, and you'll have the choice to either type a custom message or you can hit suggest which will populate a positive message or an insult.

Install tampermonkey extension, click create a new script, paste the following code, hit save and refresh HF.

```
Code // ==UserScript==
// @name      HF Quick Popularity
// @namespace  hf.tools.quickpop.stats.customboth
// @version   1.9.0
// @description Adds ?? in author statistics.
// @match     https://hackforums.net/*
// @match     https://www.hackforums.net/*
// @icon      https://hackforums.net/favicon.ico
// @grant     none
// ==/UserScript==

(function () {
    "use strict";

    /* =====
    * Settings
    * ===== */
    const POS_SUGGESTIONS = [
        "Always bringing value—appreciate your contributions!",
        "Solid help in threads. Keep being awesome!",
        "Your posts are sharp and on point. Respect.",
        "Great signal, low noise—thanks for improving the convo.",
        "Helpful and friendly—exactly what HF needs.",
        "Consistent quality. Thanks for sharing knowledge!",
        "You make tough topics easier. Kudos!",
        "Positive presence and reliable insights. Much appreciated!",
        "Great collaborator—thanks for the teamwork energy.",
        "Professional, polite, and knowledgeable. Big +rep!"
    ];

    // Insulting negatives (no slurs/threats/doxxing)
    const NEG_SUGGESTIONS = [
        "This is clueless and wastes everyone's time.",
        "Embarrassingly wrong—do your homework first.",
        "Your post reads like you didn't even try.",
```

"Painfully low-effort—adds nothing to the thread.",
"Sloppy take with zero substance.",
"You're out of your depth here.",
"Stop spreading nonsense and learn the basics.",
"Awful advice—misleading and unhelpful.",
"Feels like trolling; completely useless.",
"Brutally bad answer—hurts the discussion."

];

```
const MIN_COMMENT_LEN = 40; // server needs >=10; we ensure >=40
const FILL_DELAY_MS = 140;
const SUBMIT_DELAY_MS = 240;
```

```
// UI
```

```
const BTN_POS_TEXT = "?";
const BTN_NEG_TEXT = "?";
const BTN_CLASS = "hf-quickpop-thumb";
const ROT_POS_KEY = "hf.quickpop.rotation.pos.suggestions";
const ROT_NEG_KEY = "hf.quickpop.rotation.neg.suggestions";
```

```
/* =====
```

```
 * Utilities
```

```
 * ===== */
```

```
const delay = (ms) => new Promise(res => setTimeout(res, ms));
function toast(msg){
  const d=document.createElement("div"); d.textContent=msg;
  Object.assign(d.style,{position:"fixed",bottom:"18px",right:"18px",padding:"10px
14px",background:"rgba(0,0,0,.84)",color:"#fff",fontSize:"13px",borderRadius:"10px",zIndex:999999,
pointerEvents:"none",maxWidth:"60vw",lineHeight:"1.3"});
  document.body.appendChild(d); setTimeout(()=>d.remove(),1500);
}
function shuffle(a){for(let i=a.length-1;i>0;i--){const
j=Math.floor(Math.random()*(i+1));[a[i],a[j]]=[a[j],a[i]]}}
function pick(arr){ return arr[Math.floor(Math.random()*arr.length)]; }
function loadRotation(key, arr){
  try{
    const r=JSON.parse(localStorage.getItem(key)||"");
    if(!r||!Array.isArray(r.bucket)||!r.bucket.length) throw 0;
    return r;
  }catch{
    const idx=[...arr.keys()]; shuffle(idx);
    const r={bucket:idx,last:null}; localStorage.setItem(key,JSON.stringify(r)); return r;
  }
}
function saveRotation(key, r){ try{localStorage.setItem(key,JSON.stringify(r));}catch{} }
function nextRotated(key, arr){
  const r=loadRotation(key, arr);
  if(!r.bucket.length){ const idx=[...arr.keys()]; shuffle(idx); r.bucket=idx; }
  const i=r.bucket.shift(); r.last=i; saveRotation(key, r); return arr[i];
}
```

```
/* =====
```

```
 * Editors + syncing (input/textarea/WYSIWYG)
```

```

* ===== */
function getEditors(doc=document){
  const $ = doc.defaultView?.jQuery || window.jQuery;
  let scInst=null;
  try { if ($?.fn?.sceditor) { scInst =
$("textarea[name='comments'],textarea[name='comment'],textarea[name='notes']").sceditor?..("instance"); } } catch {}
  return {
    textInput: doc.querySelector("input[type='text'][name='comments'],
input[type='text'][name='comment'], input[type='text'][name='notes']"),
    textarea: doc.querySelector("textarea[name='comments'], textarea[name='comment'],
textarea[name='notes']") || doc.querySelector("textarea"),
    sceditorInstance: scInst,
    tinymce: (doc.defaultView?.tinymce || window.tinymce) || null,
    quillRoot: doc.querySelector(".ql-editor") || null
  };
}
function fireInputs(node){
  node.dispatchEvent(new Event("input",{bubbles:true}));
  node.dispatchEvent(new Event("change",{bubbles:true}));
  node.dispatchEvent(new Event("keyup",{bubbles:true}));
}
function syncToTextarea(doc=document){
  const ed = getEditors(doc);
  if (ed.sceditorInstance?.updateOriginal) { try { ed.sceditorInstance.updateOriginal(); } catch {} }
  if (ed.tinymce?.activeEditor) {
    try {
      const text = ed.tinymce.activeEditor.getContent({ format: "text" });
      if (ed.textarea) ed.textarea.value = text;
      if (ed.textInput && !ed.textarea) ed.textInput.value = text;
    } catch {}
  }
  if (ed.quillRoot) {
    const text = ed.quillRoot.textContent || "";
    if (ed.textarea) ed.textarea.value = text;
    if (ed.textInput && !ed.textarea) ed.textInput.value = text;
  }
}
function setNoteEverywhere(text, doc=document){
  const ed = getEditors(doc);
  try { if (ed.sceditorInstance?.val) ed.sceditorInstance.val(text); } catch {}
  try { if (ed.tinymce?.activeEditor?.setContent)
ed.tinymce.activeEditor.setContent(text.replace(/\n/g,"<br>")); } catch {}
  if (ed.quillRoot) { ed.quillRoot.innerHTML=""; ed.quillRoot.textContent=text;
fireInputs(ed.quillRoot); }
  if (ed.textInput) { ed.textInput.value = text; fireInputs(ed.textInput); }
  if (ed.textarea) { ed.textarea.value = text; fireInputs(ed.textarea); }
  syncToTextarea(doc);
}
function plainTextFromEditors(ed){
  if (ed.textInput) return (ed.textInput.value || "").replace(/\s+/g," ").trim();
  if (ed.textarea) return (ed.textarea.value || "").replace(/\s+/g," ").trim();
  if (ed.tinymce?.activeEditor) { try { return ed.tinymce.activeEditor.getContent({ format: "text"

```

```

}).replace(/s+/g," ").trim(); } catch {} }
    if (ed.quillRoot) return (ed.quillRoot.textContent || "").replace(/s+/g," ").trim();
    return "";
}
function hardenForm(container=document){
    const form = container.closest?($(".form") ||
container.querySelector?($(".form[action*='reputation.php'], form[action*='action=add']") ||
container.querySelector?($(".form"));
    if (!form || form.__hfHardened) return;
    form.__hfHardened = true;
    form.addEventListener("submit", (e) => {
        syncToTextarea(document);
        const ed = getEditors(document);
        let v = plainTextFromEditors(ed);
        if (v.length < MIN_COMMENT_LEN) {
            e.preventDefault();
            // pad with either list (neutral-ish for positives; still user's text is primary)
            while (v.length < MIN_COMMENT_LEN) v += " " +
pick(POS_SUGGESTIONS.concat(NEG_SUGGESTIONS));
            setNoteEverywhere(v, document);
            setTimeout(()=>form.submit(), 30);
        }
    }, true);
}

/* =====
 * Reputation value selection (MAX +/-)
 * ===== */
function selectMaxPositive(scope){
    const sel = scope.querySelector("select[name='reputation'], #reputation");
    if (sel) {
        let best = null;
        for (const opt of sel.options) {
            const n = Number(opt.value);
            if (!Number.isNaN(n) && n > 0) best = (best===null||n>best) ? n : best;
        }
        if (best !== null) {
            sel.value = String(best);
            sel.className = "reputation_positive";
            sel.dispatchEvent(new Event("change",{bubbles:true}));
            return true;
        }
    }
    const radios = Array.from(scope.querySelectorAll("input[type='radio'][name='reputation']"));
    if (radios.length) {
        let cand=null, candVal=-Infinity;
        for (const r of radios) { const n=Number(r.value); if(!Number.isNaN(n) && n>0 && n>candVal){
cand=r; candVal=n; } }
        if (cand) { cand.checked=true; cand.dispatchEvent(new Event("change",{bubbles:true})); return
true; }
    }
    return false;
}

```

```

function selectMaxNegative(scope){
  const sel = scope.querySelector("select[name='reputation'], #reputation");
  if (sel) {
    let worst = null;
    for (const opt of sel.options) {
      const n = Number(opt.value);
      if (!Number.isNaN(n) && n < 0) worst = (worst===null||n<worst) ? n : worst;
    }
    if (worst !== null) {
      sel.value = String(worst);
      sel.className = "reputation_negative";
      sel.dispatchEvent(new Event("change",{bubbles:true}));
      return true;
    }
  }
  const radios = Array.from(scope.querySelectorAll("input[type='radio'][name='reputation']"));
  if (radios.length) {
    let cand=null, candVal=Infinity;
    for (const r of radios) { const n=Number(r.value); if(!Number.isNaN(n) && n<0 && n<candVal){
cand=r; candVal=n; } }
    if (cand) { cand.checked=true; cand.dispatchEvent(new Event("change",{bubbles:true})); return
true; }
  }
  return false;
}

/* =====
* Compose UI (used by BOTH ? and ?)
* ===== */
function showCompose({titleText, initialText, suggestPool, submitLabel}, onSubmit, onCancel){
  const overlay = document.createElement("div");
  Object.assign(overlay.style,{
    position:"fixed", inset:"0", background:"rgba(0,0,0,.35)", zIndex:999998
  });

  const card = document.createElement("div");
  Object.assign(card.style,{
    position:"fixed", top:"50%", left:"50%", transform:"translate(-50%,-50%)",
    background:"#121212", color:"#eee", padding:"16px", width:"clamp(260px, 50vw, 520px)",
    borderRadius:"10px", boxShadow:"0 8px 30px rgba(0,0,0,.45)", zIndex:999999
  });

  const title = document.createElement("div");
  title.textContent = titleText || "Write your comment";
  Object.assign(title.style,{fontSize:"16px", marginBottom:"8px", fontWeight:"600"});

  const ta = document.createElement("textarea");
  ta.value = initialText || "";
  Object.assign(ta.style,{
    width:"100%", minHeight:"96px", resize:"vertical", background:"#1b1b1b", color:"#eee",
    border:"1px solid #333", borderRadius:"8px", padding:"10px", boxSizing:"border-box"
  });

```

```

const hint = document.createElement("div");
Object.assign(hint.style,{fontSize:"12px", opacity:.8, marginTop:"6px"});
const updateHint = () => {
  const n = ta.value.replace(/\s+/g," ").trim().length;
  hint.textContent = `Characters: ${n}${n<MIN_COMMENT_LEN?` (min ${MIN_COMMENT_LEN}
will be enforced):""}`;
};
updateHint();
ta.addEventListener("input", updateHint);

const row = document.createElement("div");
Object.assign(row.style,{display:"flex", gap:"8px", marginTop:"12px",
justifyContent:"space-between", alignItems:"center"});

const left = document.createElement("div");
const btnSuggest = document.createElement("button");
btnSuggest.textContent = "Suggest";
stylizeBtn(btnSuggest, "outline");
btnSuggest.addEventListener("click", () => {
  const s = nextRotated(suggestPool.key, suggestPool.list);
  if (!ta.value) ta.value = s;
  else ta.value = (ta.value.trim()+" "+s).trim();
  ta.dispatchEvent(new Event("input"));
});
left.appendChild(btnSuggest);

const right = document.createElement("div");
Object.assign(right.style,{display:"flex", gap:"8px"});

const btnCancel = document.createElement("button");
btnCancel.textContent = "Cancel";
stylizeBtn(btnCancel, "ghost");
btnCancel.addEventListener("click", () => { cleanup(); onCancel && onCancel(); });

const btnSend = document.createElement("button");
btnSend.textContent = submitLabel || "Send";
stylizeBtn(btnSend, "primary");
btnSend.addEventListener("click", () => {
  const text = ta.value.trim();
  cleanup();
  onSubmit && onSubmit(text);
});

right.append(btnCancel, btnSend);
row.append(left, right);
card.append(title, ta, hint, row);

overlay.addEventListener("click", (e)=>{ if(e.target===overlay){ cleanup(); onCancel &&
onCancel(); });
document.addEventListener("keydown", escClose);

document.body.append(overlay, card);

```

```

function escClose(e){ if(e.key === "Escape"){ cleanup(); onCancel && onCancel(); } }
function cleanup(){
  document.removeEventListener("keydown", escClose);
  overlay.remove(); card.remove();
}
}

function stylizeBtn(b, variant){
  b.type="button";
  const base=[
    "padding:8px 12px","font-size:13px","border-radius:8px","cursor:pointer",
    "border:1px solid transparent"
  ];
  if (variant==="primary"){
    base.push("background:#2e7d32","color:#fff","border-color:#2e7d32");
  } else if (variant==="outline"){
    base.push("background:transparent","color:#bbb","border-color:#444");
  } else {
    base.push("background:transparent","color:#888","border-color:#333");
  }
  b.style.cssText = base.join(";");
}

/* =====
* Modal helpers
* ===== */
async function waitForRepContainer(doc, timeout=4000){
  const start = Date.now();
  return new Promise(res=>{
    const check=()=>{
      const modal = doc.querySelector("#modal, .modal, .popup, .reputation_add,
form[action*='reputation.php'], form[action*='action=add']");
      if (modal) return res(modal);
      if (Date.now()-start > timeout) return res(null);
      requestAnimationFrame(check);
    };
    check();
  });
}

function findSubmit(scope){
  const btn = [...scope.querySelectorAll("button, input[type='submit']")].find(b=>{
    const t = (b.value || b.innerText || b.textContent || "").trim().toLowerCase();
    return ["submit","add","give","rate","confirm"].some(w => t.includes(w));
  });
  return btn || null;
}

function findMinLenError(doc){
  const needles = ["You must enter a comment of at least 10 characters", "at least 10 characters"];
  const errBox = doc.querySelector(".error, .errors, .errorbox, .inline_error, #error, .modal .error");
  if (errBox && needles.some(n=>errBox.textContent.toLowerCase().includes(n.toLowerCase())))
return errBox;
  if (needles.some(n=>doc.body.textContent.toLowerCase().includes(n.toLowerCase()))) return
true;

```

```

    return null;
}

/* =====
 * Thread integration (inject into author statistics)
 * ===== */
function getUidFromPost(post){
    const rate = post.querySelector("a[href^='javascript:MyBB.reputation(']");
    if (rate) { const m = rate.getAttribute("href").match(/reputation\\((\\d+)\\)/i); if (m) return m[1]; }
    const author = post.querySelector("a[href*='uid=']", a[href*='/user-'], a[href*='/member.php']");
    if (author) { try { const u=new URL(author.href, location.origin); return u.searchParams.get("uid")
|| (u.pathname.match(/user-?(\\d+)/i)||[])[1] || null; } catch{} }
    const attr = post.querySelector("[data-uid],[data-user-id]");
    if (attr) return attr.getAttribute("data-uid") || attr.getAttribute("data-user-id");
    return null;
}

function makeThumb(text, variant){ // variant: 'pos' | 'neg'
    const b=document.createElement("button");
    b.type="button"; b.textContent=text;
    b.className = BTN_CLASS + " " + (variant === "pos" ? "thumb-pos" : "thumb-neg");
    b.title = variant === "pos" ? "Give maximum positive popularity" : "Give maximum negative
popularity";
    b.style.cssText = [
        "margin-left:6px",
        "padding:2px 8px",
        "font-size:13px",
        "line-height:18px",
        "border:1px solid rgba(0,0,0,.25)",
        "border-radius:6px",
        "cursor:pointer",
        "background:" + (variant==="pos" ? "rgba(76,175,80,.12)" : "rgba(244,67,54,.12)"),
        "color:" + (variant==="pos" ? "#2e7d32" : "#c62828")
    ].join(";");
    return b;
}

function findStatsValueContainer(post){
    const rows = post.querySelectorAll(".author_statistics .author_wrapper .author_row");
    for (const r of rows) {
        const label = r.querySelector(".author_label");
        if (label && /popularity\\s*:?\\s*$/.i.test((label.textContent||"").trim())) {
            const val = r.querySelector(".author_value") || r;
            return { row: r, value: val };
        }
    }
    const wrap = post.querySelector(".author_statistics .author_wrapper");
    if (wrap) return { row: null, value: wrap, wrap };
    return null;
}

function injectInStats(post){
    if (post.dataset.hfQuickPopStatsBound === "1") return;

```

```

const spot = findStatsValueContainer(post);
if (!spot) return;

if (spot.row && spot.value) {
  const container = document.createElement("span");
  container.style.marginLeft = "8px";
  const posBtn = makeThumb(BTN_POS_TEXT, "pos");
  const negBtn = makeThumb(BTN_NEG_TEXT, "neg");
  container.appendChild(posBtn);
  container.appendChild(negBtn);
  spot.value.appendChild(container);
  wireButtons(post, posBtn, negBtn);
  post.dataset.hfQuickPopStatsBound = "1";
  return;
}

if (spot.wrap) {
  const row = document.createElement("div");
  row.className = "author_row";
  const label = document.createElement("div");
  label.className = "author_label";
  label.textContent = "Quick Pop:";
  const value = document.createElement("div");
  value.className = "author_value";
  const posBtn = makeThumb(BTN_POS_TEXT, "pos");
  const negBtn = makeThumb(BTN_NEG_TEXT, "neg");
  value.appendChild(posBtn);
  value.appendChild(negBtn);
  row.appendChild(label);
  row.appendChild(value);
  spot.wrap.appendChild(row);
  wireButtons(post, posBtn, negBtn);
  post.dataset.hfQuickPopStatsBound = "1";
}
}

function wireButtons(post, posBtn, negBtn){
  posBtn.addEventListener("click", async (e)=>{
    e.preventDefault(); e.stopPropagation();
    const uid=getUidFromPost(post); if(!uid){ toast("Can't find UID"); return; }
    showCompose(
      {
        titleText: "Write your positive comment",
        initialText: "",
        suggestPool: { key: ROT_POS_KEY, list: POS_SUGGESTIONS },
        submitLabel: "Send ?"
      },
      (userText)=>openAndSubmit(uid, "pos", userText),
      ()=>{/* cancelled */}
    );
  });

  negBtn.addEventListener("click", async (e)=>{

```

```

e.preventDefault(); e.stopPropagation();
const uid=getUidFromPost(post); if(!uid){ toast("Can't find UID"); return; }
showCompose(
{
  titleText: "Write your negative comment",
  initialText: "",
  suggestPool: { key: ROT_NEG_KEY, list: NEG_SUGGESTIONS },
  submitLabel: "Send ?"
},
(userText)=>openAndSubmit(uid, "neg", userText),
()=>{/* cancelled */}
);
});
}

```

```

function injectAll(){
  const posts = document.querySelectorAll(".post, .postbit, tr.post, div[id^='post_']");
  posts.forEach(injectInStats);
}

```

```

const mo = new MutationObserver(()=>{ if(/showthread\.php/i.test(location.href)) injectAll(); });
mo.observe(document.documentElement,{childList:true,subtree:true});
if (/showthread\.php/i.test(location.href)) injectAll();

```

```

/* =====
* Open modal, select MAX +/- , fill (uses custom text from compose), enforce, submit (+ retry)
* ===== */

```

```

async function openAndSubmit(uid, kind /* 'pos' | 'neg' */, customText /* string */){
  if (window.MyBB?.reputation) { window.MyBB.reputation(Number(uid)); }
  else { location.href = `/reputation.php?action=add&uid=${encodeURIComponent(uid)}`; return; }

```

```

const doc = document;
const container = await waitForRepContainer(doc, 4000);
if (!container) { toast("Could not open reputation dialog."); return; }

```

```

if (kind === "pos") selectMaxPositive(container); else selectMaxNegative(container);

```

```

await delay(FILL_DELAY_MS);

```

```

const base = (customText && customText.trim())
  ? customText.trim()
  : (kind === "pos" ? nextRotated(ROT_POS_KEY, POS_SUGGESTIONS) :
nextRotated(ROT_NEG_KEY, NEG_SUGGESTIONS));

```

```

setNoteEverywhere(base, doc);
hardenForm(container);

```

```

await delay(SUBMIT_DELAY_MS);
const submit = findSubmit(container);
if (submit) submit.click(); else { toast("Filled—please submit."); return; }

```

```

// Retry once if min-length error appears (pad from same pool)
setTimeout(async () => {

```

```

const err = findMinLenError(doc);
if (err) {
  let extra = base;
  const pool = (kind === "pos") ? POS_SUGGESTIONS : NEG_SUGGESTIONS;
  while (extra.replace(/s+/g, " ").trim().length < Math.max(MIN_COMMENT_LEN, 30)) {
    extra += " " + pick(pool);
  }
  setNoteEverywhere(extra, doc);
  const s = findSubmit(container);
  if (s) s.click();
}
}, 600);
}

})();

```

oRequiem - October 2, 2025, 6:14 pm

Pretty useful script, feel like Omni should add this as a feature at some point, but I still use HFX since I can just press the "+" button to add rep without leaving the page.

Charlie Sheen - October 2, 2025, 6:21 pm

(Oct 2, 2025 06:14 PM)oRequiem Wrote: [\[CITE\]](#) Pretty useful script, feel like Omni should add this as a feature at some point, but I still use HFX since I can just press the "+" button to add rep without leaving the page.

Not many still know about HFX. I know [skull](#) was considering reviving it at one point.

oRequiem - October 2, 2025, 6:27 pm

(Oct 2, 2025 06:21 PM)Charlie Sheen Wrote: [\[CITE\]](#) Not many still know about HFX. I know [skull](#) was considering reviving it at one point.

I was thinking about it since I have the files, but would need adam's permission I would assume. I just made it so it works on chrome after they nuked manifest v2.

Charlie Sheen - October 2, 2025, 6:30 pm

(Oct 2, 2025 06:27 PM)oRequiem Wrote: [\[CITE\]](#) I was thinking about it since I have the files, but would need adam's permission I would assume. I just made it so it works on chrome after they nuked manifest v2.

What's his brothers name bennetxk or something? He gave [skull](#) permission and offered to help if needed.

oRequiem - October 2, 2025, 6:30 pm

(Oct 2, 2025 06:30 PM)Charlie Sheen Wrote: [\[CITE\]](#) What's his brothers name bennetxk or something? He gave [skull](#) permission and offered to help if needed.

I wouldn't know haha, but ah okay then skull could cook with it

Charlie Sheen - October 2, 2025, 6:35 pm

(Oct 2, 2025 06:30 PM)oRequiem Wrote: [\[CITE\]](#) I wouldn't know haha, but ah okay then skull could cook with it

<https://hackforums.net/member.php?action=profile&uid=574588>

He can get ahold of Adam. I'd like to see HFX return and would help in any way I could.

6 - October 2, 2025, 7:12 pm

Great contribution as always.

oRequiem - October 2, 2025, 7:53 pm

(Oct 2, 2025 06:35 PM)Charlie Sheen Wrote: [\[CITE\]](#)
<https://hackforums.net/member.php?action=profile&uid=574588>

He can get ahold of Adam. I'd like to see HFX return and would help in any way I could.
Should be easy once you have the files, just need to make it manifest v3 really with some upgraded features like what your script does.

Spore - October 2, 2025, 8:42 pm

Do you have any screenshots of it? Curious how the thumbs up/down look within the postbit.

Stanley - April 24, 2026, 9:16 pm

I'm Stanley HF's helper bot. The OP (original poster) of this thread wants everyone to see this page and has elected to bump it to the top of the showthread listing. Since HF doesn't want a ton of LQP (Low Quality Posts) this is the best method of bumping your content. There is a small ?yte fee for each bump. Any OP can bump any of their threads on HF from the linked icon on the top of their first post.

If you haven't already read the first post of this thread we encourage you to do so.

Thanks,
Stanley